GenRad

Development Systems Division

MICROCOMPUTER
SYSTEMS

GENRAD/DEVELOPMENT SYSTEMS DIVISION

PROGRAMMING THE AMDS

MANUAL NO. 2300-5001-00

GENRAD/DEVELOPMENT SYSTEMS DIVISION
5730 BUCKINGHAM PARKWAY
CULVER CITY, CA 90230

PREFACE

This manual provides information on the AMDS memory structure, utility subroutines, and direct I/O programming, and is organized in three sections as follows:

1) "Debugger Memory Usage" describes memory configuration and the essentials of Debugger operation.
2) "I/O Service Subroutines" explains the calling conventions and operation of the system I/O routines contained on the AMDS Utility Source Package diskette. The I/O subroutine calls provide access to system resources.
3) "Direct I/O Programming" provides a detailed discussion of the AMDS I/O structure. This information is helpful for the user who wishes to write custom utility programs for interfacing with specialized devices.

TABLE OF CONTENTS

Section		Page
1	Programming the AMDS	1
	MEMORY CONFIGURATION	1
	DEBUGGER INTERRUPT VECTOR USAGE	2
	DEBUGGER USE OF THE CPU STACK	3
2	I/O Service Subroutines	4
	KEYBOARD INPUT/CRT OUTPUT	6
	Keyboard and CRT Display Functions	7
	MSGIN	7
	MSGOUT	7
	KEYIN	7
	TVOUT	7
	DISPCUR	7
	OUTCUR	8
	BLNKCUR	8
	KBDIN	8
	KBDTST	8
	DISK INPUT/OUTPUT	9
	File Control Block	9
	Diskette Directory	19
	Error Return Codes	20
	Function Definitions	22
	Open	22
	Read	22
	Write	22
	Close	24
	Create	24
	Delete	24
	Rename	24
	Change Attributes	24
	Free Space	24
	Overlapped I/O	25
	Examples	27
	Reading From a File	27
	Writing Into a File	28

TABLE OF CONTENTS (Concluded)

Section		Page
	EIA INPUT/OUTPUT	31
	Definitions	31
	EIASET	31
	EIAIN	31
	EIAOUT	31
	ECBRATE	32
	ECBLEN	32
	ECBUNIT.	32
	Examples	33
	ASCII Data Configuration	33
	ECB Communication	33
	CASSETTE TAPE INPUT/OUTPUT	35
	Error Recovery	35
	WRITE	35
	READ	37
	READR	39
	Cassette Rewind Routine	39
3	Direct I/O Programming	40
	KEYBOARD INPUT	41
	BREAK KEY	42
	CRT DISPLAY SELECT	43
	CRT DISPLAY OUTPUT	44
	MEMORY PROTECTION	46
	REAL-TIME CLOCK	48
	SERIAL I/O PORTS	49
	DISK I/O PORT	55
	PRINTER PORT	56
	BOARD STATUS PORT	57
	BOARD COMMAND PORT	58
	JUMPER SELECTION	59
	AMDS I/O PORT ADDRESSES	61
APPENDIX		

LIST OF TABLES

Table		Page
1-1.	Debugger Address and Vector Allocation	2
2-1.	I/O Service Subroutine Functions	5
2-2.	CRT Display Control Command Functions	6
2-3.	Disk File Control Block Fields	9
2-4.	Disk File Management Functions and FCB Fields	12
2-5.	Disk File Management Error Return Codes	21
2-6.	EIA Send-Control Block	32
2-7.	EIA Baud Rate Select Codes	33
2-8.	Tape Write Control Block	36
2-9.	Tape Read-Control Block	38
2-10.	Tape Read Routine Error Codes	38
3-1.	Keyboard Input Commands	41
3-2.	BREAK Key Commands	42
3-3.	CRT Display Select	43
3-4.	CRT Field Attributes	44
3-5.	CRT Display Line Offsets	45
3-6.	Protect Command Bit Assignments	46
3-7.	Memory Protect Commands	46
3-8.	Real-Time Clock Commands	48
3-9.	Serial Port 1 Read Commands	49
3-10.	Serial Port 1 Write Commands	51
3-11.	Serial Port 2 Read Commands	52
3-12.	Serial Port 2 Write Commands	53
3-13.	Disk I/O Port Commands	55
3-14.	Printer I/O Commands	56
3-15.	Board Status Port Commands	57
3-16.	Board Command Port Commands	58
3-17.	Jumper Functions	59
3-18.	Summary of I/O Port Addresses	61

Section 1

Debugger Memory Usage

MEMORY CONFIGURATION

The AMDS normally contains 48K bytes of read/write memory, usually occupying the low-order memory addresses from X'0000' to X'7FFF', and from X'A000' to X'DFFF'. The range from X'A000' to X'BFFF' is also mapped into the area from X'8000' to X'9FFF'. The bootstrap loader uses the high-order 8K-byte area: locations X'E000' to X'FFFF' for 8080/85A/Z80 and X'C000' to X'DFFF' for 6800/02 systems. For 6800/02 systems, the area from X'E000' to X'FFFF' contains random access memory (RAM).

The bootstrap loader occupies a 1024-byte PROM which responds to all of the addresses mentioned above. There are no restrictions on referencing memory locations which are not implemented. When a non-implemented location is read, the value X'FF' is obtained.

Memory mapping is structured so that the interrupt vectors (from X'0000' to X'0069' for 8080/85A/Z80; X'FFF8' to X'FFFF' for 6800/02) exist in RAM, thus permitting user modification.

The 6800/02 I/O instructions refer to a portion of the main memory address space. The 6800/02 I/O ports are located in the address spaces from X'D000' to X'DFFF'. Since these I/O addresses are not completely decoded, the AMDS uses 4,096 address locations to provide 256 I/O ports.

The 8080, 8085A and Z80 I/O instructions refer to a separate address space. See Table 3-18 for a list of I/O addresses used by the AMDS.

DEBUGGER INTERRUPT VECTOR USAGE

The 8080, 8085A and Z80 have a common 8-level interrupt structure, while the 6800 and 6802 have a 4-level interrupt structure which includes Reset and Software Interrupt. The Debugger uses the area of memory from X'0000' to X'0069' for 8080/85A/Z80 vector interrupts, and locations 'FFFA' to X'FFFE' for 6800/02 vector interrupts (see Table 1-1, below).

Table 1-1. Debugger Address and Vector Allocation

Function	8080	8085	Z80	6800/02
RESET	X'0000'	X'0000'	X'0000'	X'FFFE'
BREAK	X'0020'	X'2C'	X'66'	X'FFFC'
Debugger Breakpoint	X'0030'	X'2C'	X'66'	X'FFFC'
Software Interrupt				X'FFFA'

If these locations are changed, the Debugger features associated with them will be disabled. The Debugger functions can be re-enabled using the RESTORE function.

For information on the assignment of vector addresses to I/O device interrupts (other than Debugger), refer to Section 3.

NOTE: The Z80 CPU automatically masks all internal interrupts in hardware. To enable internal interrupts (for use of the real-time clock facility in the BASIC Interpreter, for example), an X'04' must be written to I/O port X'F6'.

DEBUGGER USE OF THE CPU STACK

Upon entry to the Debugger, the stack pointer is set to a valid (non-protected RAM) address. When program execution is interrupted by the BREAK key or a breakpoint instruction, control is transferred to the Debugger which stores the machine status and registers on the stack and saves the stack pointer address. The user may or may not reset the stack pointer after the interrupt; Debugger execution will continue automatically in either case. Note, however, that if the user resets the stack pointer to an invalid address, the machine status display may show incorrect data.

The Debugger Execute command (E) stores the starting address on the stack, reloads the registers, and executes a return instruction. The return instruction transfers control to the execute address which is stored on the stack. If the stack pointer is not valid, the execute command will fail and the message "SP NOT IN RAM" will be displayed.

The "Z S=n" command is used to reset the stack pointer. Since the Debugger generates the machine status display from the data on the stack, the display changes each time the "Z S=n" command is entered.

Upon Debugger entry or upon re-entry (after hitting the RESET key), the stack pointer is set to the first byte in memory below the Debugger. When a program is executed using the "E" command, this value is loaded into the stack pointer register. The user may use this value or may load a new one.

Section 2
I/O Service Subroutines

Relocatable subroutines, including a number of I/O service subroutines, are available to the user. Service subroutines greatly simplify input/output operations to standard peripheral devices. These subroutines may be actuated by execution of a CALL instruction (for 8080/85A/Z80) or JSR instruction (6800/02) to the appropriate entry point, as shown in Table 2-1.

I/O subroutines are provided for keyboard input/CRT display output, disk input/output, EIA send/receive, and cassette tape read/write functions. The routines have convenient calling conventions for use with the I/O devices. The routines interpret user requests and perform the detailed low-level I/O operations required to accomplish the requested operations.

The I/O service subroutines are listed in Table 2-1 and are described in detail on the following pages.

Table 2-1. I/O Service Subroutine Functions

I/O Type	Global Label	File	Service Function
Keyboard/ CRT Display	KEYIN	KEYIN.R	Read a character from the keyboard or command file
	MSGOUT	KIO.R	Write a message a line at a time to the CRT display
	MSGIN		Read and echo a line from the keyboard
	TVOUT		Write a character to the CRT display
	DISPCUR		Display the cursor
	OUTCUR		Display the cursor
	BLNKCUR		Blank the cursor
	KBDIN		Read a character from the keyboard
	KBDTST		Test for keyboard input
Disk	DISK	DIO.R	Entry to the disk routines
	DISKW		Wait for completion of disk I/O
EIA	EIASET	EIA.R	Set EIA parameters
	EIAIN		Input a character from the EIA port
	EIAOUT		Send a character to the EIA port
Tape	READ	TIO.R	Read a record from tape
	READR		Re-entry read from tape
	WRITE		Write a record to tape
	REWIND		Rewind tape Unit(s)

KEYBOARD INPUT/CRT DISPLAY OUTPUT

The keyboard and CRT display terminal may be addressed directly (see Section 3). However, a much easier method of accessing them is obtained by use of MSGIN and MSGOUT which access a line at a time; KEYIN and TVOUT which access a character at a time; the cursor display routines (DISPCUR, OUTCUR, and BLNKCUR); and the bypass command file routines (KBDTST and KBDIN).

The TVOUT routine operates the CRT display in the following manner. The cursor is first initialized to the top left display location by outputting a clear (DLE=X'10') character. The ASCII representation of the character to be displayed is then loaded into the accumulator* prior to calling the TVOUT routine. A cursor is displayed at the next character position. When that character is stored, the cursor is advanced, moving from the last character of the line to the first character of the following line. When the last character of the bottom line is reached, it moves to the first character of the top line.

A horizontal dividing line below the cursor (consisting of underline (_) characters) separates the data most recently displayed (above the line) from older data (below the line). As new data are received and entered onto the display, the dividing line moves down. At the bottom of the screen it wraps around to the top so that the data remain on the screen until they are written over on the next cycle. All registers and flags are saved.

The most commonly used control commands are interpreted as shown in Table 2-2.

Table 2-2. CRT Display Control Command Functions

Character	Function
Carriage Return	Moves the cursor to the left of the current line.
Line-Feed	Moves the cursor down one line.
Backspace	Moves the cursor back one space.
DLE(X'10')	Clears the screen and leaves the cursor at the top left corner.

* In this document the word "accumulator" refers to the "A" register in 8080/85A/Z80 systems and to accumulator "A" in 6800/02 systems.

Keyboard and CRT Display Functions. The nine keyboard CRT I/O functions are defined below.

MSGIN

MSGIN reads a line at a time from the keyboard or command file and echoes the data on the display. MSGIN uses routines in the KEYIN.R, LINEIN.R and KIO.R files.

MSGOUT

Displays a line at a time. When MSGOUT is called, a message is written to the CRT display refresh page. The address of the message is passed in the HL register pair (or Index register for 6800/02 systems). The message is terminated by X'00'.

KEYIN

KEYIN reads a character at a time from the keyboard or command file and returns it in the accumulator. If the accumulator is zero when the routine is called, the character is converted to upper case; if non-zero, lower case characters can be read. The Command File Processor and KEYIN use memory from X'D600' to X'D7FF' for the command file workspace. The user should not alter this region if KEYIN is used. KEYIN uses the LINEIN routine (in file LINEIN.R) and routines in file DIO.R.

TVOUT

Displays a character at a time. The accumulator must contain the character to be displayed; no other parameters are used.

DISPCUR

When called, this routine displays the cursor. If the content of the HL register pair/Index register is zero, the cursor is displayed at the next location to be used by TVOUT. If the HL register pair/Index register has a non-zero value, this value is assumed to be a display refresh page address at which the cursor will be displayed. This entry point assumes there are no field attribute characters in the display refresh page.

OUTCUR

OUTCUR is used to display the cursor when field attribute characters are present in the display refresh page (see Section 3) or when the cursor location is in row-column form. At the time of invocation, the accumulator should contain the column number and the B register/accumulator B should contain the row number.

BLNKCUR

This routine blanks the cursor. No parameters are used.

KBDIN

KBDIN reads a character from the keyboard and returns the result in the accumulator. This routine can be used to read from the keyboard even if command file processing is active.

KBDTST

This routine tests the keyboard status register. If new data are available, the accumulator will be non-zero on return. If a keystroke has not been detected, the accumulator will be zero.

DISK INPUT/OUTPUT

FUTUREDATA RDOS allows user programs to implement all of the disk file management functions. These functions (Create, Delete, Open, Close, Read, Write, Rename, Change Attributes, and Free Space) are made available through a single entry point in the relocatable disk I/O package DIO (residing in file NDIO.R).

File Control Block. The Disk I/O service routines are invoked by executing a call to the entry point, DISK. The instruction, CALL DISK (or JSR DISK for 6800/02 systems), invokes the disk I/O package. The address of a 26-byte parameter list (see Table 2-3) which is passed in the HL register pair/Index register, completely specifies the operation to be performed. This list is called a File Control Block (FCB), and should be built in any convenient area in read/write memory. The FCB is used to determine the required function, file name, location of the I/O buffers, etc.

The Disk I/O routines return a completion code in the accumulator to indicate whether the operation was performed successfully, or if an abnormal condition was detected. When errors are detected, the HL register pair/ Index register points to the address of an ASCII message giving the nature of the error. This message is terminated by a byte of zero and may be displayed by calling the MSGOUT I/O service subroutines.

The 13 fields contained within the FCB are listed in Table 2-3. Table 2-4 shows the file management functions and how they relate to the FCB fields.

Table 2-3. File Control Block Fields

Field	Bytes	Function
CBFLO	1	X'00' - perform function defined by FCBFL1 X'02' - create file X'04' - open file X'08' - close file X'10' - delete file X'20' - rename file X'40' - change file attributes X'80' - free unused space

Table 2-3. File Control Block Fields (Continued)

Field	Bytes	Function
FCBFL1	1	Read-Write Functions Bit* Function 0 Read Sector 1 Write Sector 2 Perform Write Check 3 Unused 4 Automatically increment sector number 5 Read or Write Object Data 6 Unused 7 Do not wait for completion
FCBADR	2	Address of data block in main memory to be transferred to (from) diskette. For the 8080/85A/Z80, the address is in low-high format; for the 6800/02 the address is in high-low format.
FCBLEN	2	Length of data block in main memory to be transferred to (from) diskette. For 8080/85A/Z80, the length is in low-high format; high-low format for 6800/02. The maximum allowable length is the sector size of the diskette, 128 bytes.
FCBSEC	2	Specifies the sector to be read or written. The first sector is designated as 1. The nth sector relative to the beginning of the file is designated as "n". For 8080/85A/Z80 this parameter is stored in low-high format; high-low format for 6800/02.
FCBUN	1	Selects drive 0 (X'00') or drive 1 (X'01').

*Low-order (rightmost) bit is zero.

Table 2-3. File Control Block Fields (Concluded)

Field	Bytes	Function
FCBDIR	1	Directory entry location. This field is set and used by the Disk I/O routines.
FCBNAM	10	Alphanumeric file name, left-justified in the field, i.e., if the file name is less than 10 characters long, it must be followed by a sufficient number of space characters (X'20') to fill the field.
FCBREF	1	First track of file. This field is set and used by the Disk I/O routines.
CBATR	1	File attributes. Bit — Function 0 — Write Protect 1 — Permanent File 2 — Source File 3 — Object File 4 — Relocatable File 5 — Command File 6 — Blind File 7 — System File
FCBEOF	2	Specifies end of file (one greater than the number of sectors within a file). An empty file is shown as X'0001'. This field is stored in a normal high-low format. When a file is opened for output, the high order bit is set; the bit is reset when the file is closed.
FCBINT	1	Initial file size, specified in tracks. This field is required only for Create. The minimum size is 1 (X'01'), and the maximum size is 76 (X'4C').
FCBEXT	1	Extension size in tracks. Used when file must expand. Allowable values are the same as the initial allocation.

Table 2-4. File Management Functions and FCB Fields

FCBFL0 - 1 byte			FCBFL1 - 1 byte
Function	Value	Function Description	Function Description
Open	X'04'	Selects the Open function.	Bits 0* and 1 are examined to determine whether file is to be opened for reading or writing, respectively.
Read	X'00'	Causes function specified by FCBFL1 to be performed.	Bit 0 is set to designate a Read operation. Bit 4 is optionally set to indicate sequential reading of the file.
Write	X'00'	Causes function specified by FCBFL1 to be performed.	Bit 1 is set to perform a Write. Bit 2 is set to perform a Write Check. Bit 4 specifies that FCBSEC is automatically incremented before each Write for writing sequentially.
Close	X'08'	Selects Close function.	Bits 0 and 1 are set to indicate that the file was opened for reading or writing, respectively.
Create	X'02'	Selects Create function.	Bits 0 and 1 are set to indicate that the file was opened for reading or writing, respectively.
Delete	X'10'	Selects Delete function.	
Rename	X'20'	Selects Rename function.	
Change Attributes	X'40'	Selects Change Attributes function.	
Free Space	X'80'	Selects Free Space function.	

*Bit zero is the rightmost bit.

Table 2-4. File Management Functions and FCB Fields (Continued)

Function	FCBADR - 2 bytes Function Description	FCBLEN - 2 bytes Function Description
Open		
Read	Gives address of buffer to which data is to be transferred. Stored in low-high (8080/85A/Z80) or high-low (6800/02).	Specifies number of bytes in buffer to be transferred, maximum of 128 bytes. Stored in low-high format (8080/85A/Z80); high-low (6800/02).
Write	Specifies the address from which data for Write operations are to be obtained. Stored in low-high (8080/85A/Z80); high-low (6800/02).	Indicates size of data area to be transferred. Low-high format (8080/85A/Z80); high-low (6800/02).
Close		
Create		
Delete		
Rename		
Change Attributes		
Free Space		

Table 2-4. File Management Functions and FCB Fields (Continued)

Function	FCBSEC - 2 bytes Function Description	FCBUN - 1 byte Function Description
Open		Specifies disk drive to be accessed.
Read	Specifies number of sector to be read. Stored in low-high format (8080/85A/Z80); high-low (6800/02).	Drive number; field set prior to Open.
Write	Designates number of sector to be written. Low-high (8080/85A/Z80); high-low (6800/02).	Field set prior to Open.
Close		Field set prior to Open.
Create		Selects disk drive to be accessed.
Delete		Field set prior to Open.
Rename		Field set prior to Open.
Change Attributes		Field set prior to Open.
Free Space		Field set prior to Open.

Table 2-4. File Management Functions and FCB Fields (Continued)

Function	FCBDIR - 1 byte Function Description	FCBNAM - 10 bytes Function Description
Open		Specifies 10-character name of file to be opened.
Read		
Write	Field set by Open.	
Close	Field set by Open.	Field set prior to Open.
Create		Specifies name of file to be created.
Delete	Field set by Open.	
Rename	Field set by Open.	Contains new file name.
Change Attributes	Field set by Open.	Field set by Open.
Free Space	Field set by Open.	Field set by Open.

Table 2-4. File Management Functions and FCB Fields (Continued)

Function	FCBREF - 1 byte Function Description	FCBATR - 1 byte Function Description
Open		
Read	Field set by Open.	
Write	Field set by Open.	Field set by Open.
Close	Field set by Open.	Field set by Open. May be reset prior to Close if attributes are to be changed.
Create		Indicates which attributes are to be initially assigned to the new file.
Delete	Field set by Open.	Field set by Open.
Rename	Field set by Open.	Field set by Open.
Change Attributes	Field set by Open.	Field set by Open. May be updated prior to execution of Change Attributes function.
Free Space	Field set by Open.	Field set by Open.

Table 2-4. File Management Functions and FCB Fields (Continued)

Function	FCBEOF - 2 bytes Function Description	FCBINT - 1 byte Function Description
Open		
Read	Field set by Open.	
Write	Field set by Open.	
Close	Field set by Open and updated during Write.	
Create		Number of tracks to be allocated to file.
Delete		
Rename	Field set by Open.	
Change Attributes	Field set by Open. May be updated prior to execution of Change Attributes function.	
Free Space	Field set by Open.	

Table 2-4. File Management Functions and FCB Fields (Concluded)

Function	FCBEXT - 1 byte Function Description
Open	
Read	
Write	Field set by Open.
Close	Field set by Open.
Create	Number of tracks to be taken during a Write when additional space is required.
Delete	
Rename	Field set by Open.
Change Attributes	Field set by Open.
Free Space	Field set by Open. Used to determine amount of space used in file.

Diskette Directory. All diskettes used with FUTUREDATA RDOS must have a directory located on physical track zero. The directory is created when the diskette is initialized with the "I" command in the File Manager. The file name "DIR" is entered for the directory. DIR is given the "permanent" and "write protect" attributes. The purpose of the directory is to store the name of each file, its location on the diskette, its attributes, its current end-of-file location, and size of expansion when additional space is required.

The directory is made up of one sector containing a track map and ten sectors containing file name information. The track map maintains a list of both unallocated tracks and those allocated to existing files. The file name sectors contain one entry for each file allocated on the diskette. Each name entry is 16 bytes long. Eight entries fit into each 128-byte sector; thus, ten sectors can accommodate a maximum of 80 entries, with a maximum of 76 usable sectors.

The following FCB fields are contained in each file name entry: FCBNAM (10 bytes), FCBREF (1 byte), FCBATR (1 byte), FCBEOF (2 bytes), FCBINT (1 byte), and FCBEXT (1 byte).

The file name entries are initialized with the Create function, removed with the Delete function, and changed with the Rename function. The file attributes, end-of-file, and expansion size can be modified with the Change Attributes function.

Error Return Codes. A completion code is loaded into the accumulator upon return from the Disk I/O routines. This code indicates whether the requested function executed successfully, or whether an error condition was encountered. The accumulator is set to zero if no errors have been found. If an error has been detected, an error code is placed in the accumulator, and the HL register pair/Index register is set to point to an ASCII message describing the error. The error message is stored in memory as a sequence of ASCII characters followed by a zero byte indicating the end of the message. The error message may be displayed by the calling program.

The I/O error messages and the data file management functions that generate them are shown in Table 2-5.

Table 2-5. Disk File Management Error Return Codes

I/O Routine Error Messages		Disk File Management Functions								
Return Code	Definition	Open	Read	Write	Close	Create	Delete	Rename	Change Attributes	Free Space
0	Function completed successfully.	X	X	X	X	X	X	X	X	X
1 - PERM I/O ERR	Unrecoverable error.	X	X	X	X	X	X	X	X	X
2 - END FILE	Reading of a sector beyond the end-of-file was requested.		X							
3 - DISK FULL	No space is available to expand the size of the file.			X		X				
4 - FILE NOT FOUND	Specified file was not found in diskette directory.	X								
5 - DUPLICATE NAME	File name specified is already used on the diskette.					X		X		
6 - PARM ERR	Invalid field in FCB.	X	X	X	X	X	X	X	X	X
7 - DRIVE NOT UP	Selected disk drive not ready.	X	X	X	X	X	X	X	X	X
8 - PERM FILE	File requested for deletion has the permanent attribute.						X			
9 - WRITE PROTECT	Attempt was made to open a file with the Write Protect attribute.	X		X						
10 - FILE NOT OPEN	File has not been opened.		X	X	X		X	X	X	X

Function Definitions. The file management functions are described in the paragraphs below.

Open Function. Before a file can be accessed on diskette, it must be "opened". The Open function uses the ASCII file name given in the FCB to locate the file name entry in the diskette directory. This entry provides the information necessary for file access such as location on the diskette, end-of-file, etc.

Read Function. The Read function causes the disk unit to seek the requested file sector. Data from the sector are then transferred into the memory buffer specified by the FCBADR field of the FCB. The diskette sectors contain 128 bytes; the length to be transferred to memory (up to 128 bytes) can be explicitly specified in the FCB.

File sectors are numbered sequentially, starting with the number 1. The sector to be read is contained in the FCBSEC field of the FCB. Even though a file may be composed of a number of non-contiguous tracks, the Disk I/O routines automatically compute the location of the requested sector.

When a file is first opened, the FCBSEC field is set to zero. If direct access to file sectors is required, the program must enter the sector numbers in the FCBSEC field prior to each read request. If sequential accessing to file sectors is desired, bit 4 of the FCBFL1 field may be set so that FCBSEC will be incremented by the Disk I/O routines. The first Read request reads the first sector of the file, the second request reads the second sector, etc. Bit 4 specifies that the current value of the FCBSEC field will be automatically incremented by one, prior to each Read request.

Prior to each Read operation, the Disk I/O routines compare the requested sector number with the end-of-file sector number in the FCBEOF field. If a sector beyond the end-of-file is requested, the Read function returns with code in the accumulator specifying an "end-file" condition.

Write Function. Writing to a diskette sector consists of two physical operations. First, the data from memory are written to the diskette. That sector is then read to insure that the data have been successfully written. This secondary Read is known as a "Write Check" operation.

The RDOS Disk I/O routines allow the Write and the Write Check to be performed either by one call, or by separate calls. Bits 1 and 2 of the

FCBFL1 field specify which operations are to be performed. Normally, both bits 1 and 2 should be set so that the Write and Write Check operations will be performed. However, if both operations are performed by one call, there is a delay of one diskette revolution (167 ms) between the Write and the Write Check, since the same sector must be accessed twice.

In some circumstances, it is possible to design a program so that a large amount of data can be written sequentially from memory. In this case, a significant amount of write time can be saved if all of the Write operations are performed first. FCBSEC must then be reset to the first written sector to perform the Write Checks.

When the Write and Write Check operations are performed separately, it is possible to write five sectors per revolution (30 sectors per second) and to Write Check five sectors per revolution.

As with the Read function, the file must be opened before a Write operation can occur. The Write function requires the FCBADR field (specifying the address) and the FCBLEN field (specifying the file size) to define the memory buffer to be written to diskette.

FCBSEC designates the number of the sector to be written into. FCBSEC is initialized to zero when the file is first opened. For direct writing, FCBSEC is set by the program prior to initiating the Write operation. For sequential writing, bit 4 of FCBFL1 must be set; the Disk I/O routines will then automatically increment FCBSEC prior to each Write operation.

Before writing to the diskette, the Disk I/O routines check that the sector number specified by FCBSEC is less than the end-of-file reference in FCBEOF. Before a sector is sequentially written, FCBSEC is incremented by one. If FCBSEC becomes greater than FCBEOF, the Disk I/O routines attempt to allocate more space for the file. The number of tracks obtained when an allocation extension is made is given by the value in FCBEXT. Sector allocation may continue until free space on the diskette is unavailable.

When the user wishes to shorten an existing file, the FCBEOF field must be reset to a value one sector greater than the number of the last used sector. When a previously written file is to be rewritten sequentially, the FCBEOF field should be reset to X'0001' immediately after the file is opened. The field will then be automatically incremented to keep track of the new end-of-file.

Close Function. The Close function updates the diskette directory with new or modified information, such as file attributes and end-of-file location. If a file has been opened for reading only and no changes were made to its directory entry (this is the usual case), the Close function is not manditory. However, if the file has been opened for writing, it is recommended that the file be closed since there is a possibility that information about the file or file length has changed.

The Close function can be used to set or change file attributes. This is accomplished by changing the FCBATR field prior to execution of the function.

Create Function. The Create function is used to set up new files on a diskette. Create automatically opens the file so that other functions can be performed directly after creation.

The size of the new file is specified by the FCBINT field. When the file is created, the attributes specified in FCBATR and the extension size specified in FCBEXT are stored in the file's diskette directory entry for later use.

Delete Function. The Delete function is used to erase an existing file, return the previously allocated space to the free track pool, and remove the file's directory entry.

Rename Function. The Rename function is used to change the name of an existing file.

Change Attributes Function. The Change Attributes function allows the attributes of an existing file to be modified. The following FCB fields may be affected by this function: the attribute field (FCBATR), the end-of-file field (FCBEOF), and the expansion size field (FCBEXT).

Free Space Function. The Free Space function can be used to return unused tracks to the free track pool. When an existing file is rewritten, the new data may occupy less space than the data originally stored in the file. The Free Space function uses the current end-of-file, as specified by the FCBEOF field, and returns all tracks not needed by the new file to the free track pool. If all of the tracks are used, the Free Space function makes no changes.

Overlapped I/O. The Disk I/O routines provide for processor execution overlapped with disk reading or writing. Overlapped I/O is selected by setting the high-order bit of the FCBFL1 field. When overlapped I/O is specified, the Disk I/O routines return to the calling program upon execution of the disk Read or Write commands. To check for completion of the requested operation, the user program must call the Disk routines a second time.

When a Read operation is desired, the first call to the Disk I O routines checks the parameters, starts the read function, and returns control to the calling program. The read operation transfers data from diskette into a sector-sized buffer contained in the disk control unit. The second call to the Disk routines checks for completion of the Read. If the Read operation has not been completed, the program waits. If the operation has been completed, the data are transferred from the control unit buffer into the buffer specified in the FCB.

The Write operation is begun by the Disk routines' transfer of data from the buffer specified in the FCB to the disk control unit sector buffer. The Disk routines return to the calling program, and a second call is made to the Disk routines. If the Write operation has not been completed, the program waits. If the Write has been completed, the Disk routines return to the calling program.

The Write Check process is similar to that of the Write operation. When overlapped I/O is selected, only one operation can be performed by a single call to the Disk I/O routines. If both the Write and Write Check functions are selected, only the Write operation will be performed.

The first call to the Disk routines, which initiates overlapped I/O, is performed in the same fashion as calls for non-overlapped functions. For the 8080/85A/Z80, the HL register pair contains the address of the FCB, and the Call instruction references the Disk I/O entry point (DISK). For the 6800/02, the Index Register contains the address of the FCB, and the JSR instruction references the Disk I/O entry point (DISK).

The second call to the Disk routines, which completes the function, references a secondary entry point (DISKW). The wait entry does not require that the FCB address be loaded into the HL register pair/Index register, since the FCB referenced when the operation was initiated is used. However, the

accumulator must be set to indicate one of the following actions: 1) if the accumulator is set to zero, normal processing will occur as described above; 2) if the accumulator is set to non-zero (e.g., X'01'), the I/O operation in progress will be terminated immediately.

Since both disk drives are accessed through the same control unit, it is not possible to perform concurrent operations to both disk drives. Thus, if the second disk drive must be accessed while operation is in progress on the first, one of two actions must be taken: 1) the operation on the first drive must be completed by a call with the accumulator set to zero, or 2) the operation must be aborted by a call with the accumulator set to non-zero. In the latter case, the operation must be restarted if the data are needed.

Return codes are generated by both the first and second calls to the Disk routines. An error check must be made following both the initial call and the call completing the operation. If an I/O error is detected during the second call to the Disk routines, automatic error recovery is attempted. The operation is re-tried ten times without returning to the calling program. During error recovery, overlapped execution is temporarily suspended. If the error is recoverable, the return code indicates successful completion; if not, the I/O error return code is generated.

Examples. The following paragraphs contain example sequences for creating, opening, reading, writing, and closing files using the 8080/85A/Z80. When using the 6800/02, two-byte fields in the File Control Block (B-type addresses) must be in high-low format (A-type addresses).

Reading From a File. In order to read from an existing file, the file must be opened. This is accomplished by calling the Disk routines with the Open function selected by an X'04' in the first byte of the FCB. In the example below, the FCB has the symbolic name "RFCB". RFCB is set up in memory using Define Constant (DC) Assembler directives. The fields required by Open and Read are initialized in RFCB.

The instruction sequence for opening the file is as follows. Note: the GLBL Disk directive must occur once (and only once) in an assembly that calls the Disk I/O routines.

```
    (GLBL DISK                  EXTERNAL ENTRY POINT)
    LXI H,RFCB                  LOAD H,L WITH FCB ADDR
    MVI M,X'04'                 SELECT OPEN FUNCTION
    CALL DISK                   CALL DISK ROUTINE
    ORA A                       CHECK RETURN CODE
    JNZ ERROR                   HANDLE ERROR RETURN
    MVI M,0                     SELECT READ FUNCTION FOR LATER
```

Sectors from the open file can be sequentially read by the following:

```
    (GLBL DISK                  EXTERNAL ENTRY POINT)
    XI H,RFCB                   LOAD H,L WITH ADDR OF RFCB
    CALL DISK                   CALL DISK ROUTINE
    ORA A                       CHECK RETURN CODE
    JNZ ERROR                   HANDLE ERROR CONDITION
```

The File Control Block for the above operations is as follows:

```
RFCB DC  0                      FUNCTION IS SET LATER
     DC  X'11'                  SELECT READ & INCREMENT
     DC  B(IBUF)                ADDR OF READ BUFFER
     DC  B(128)                 LENGTH OF READ BUFFER
     DC  B(0)                   SECTOR # INIT BY OPEN
```

```
      DC  0                          SELECT UNIT ZERO
      DC  0                          FIELD SET BY OPEN
      DC  'FILENAME  '               FILE NAME
      DC  0                          FIELD SET BY OPEN
      DC  0                          ATTRIBUTE SET BY OPEN
      DC  A(0)                       EOF SET BY OPEN
      DC  0                          ALLOCATE SET BY OPEN
      DC  0                          EXTENSION SIZE SET BY OPEN
IBUF  DS  128                        READ BUFFER
```

Writing Into a File. The example below shows the procedure for creating and writing into a file. First, the file is created with the following sequence:

```
      (GLBL DISK                     EXTERNAL ENTRY POINT)
      LXI   H,WFCB                   LOAD FCBADDR
      MVI   M,X'02'                  SELECT CREATE
      CALL  DISK                     CALL DISK ROUTINE
      ORA   A                        CHECK RETURN CODE
      JNZ   ERROR                    HANDLE ERROR CONDITION
      MVI   M,0                      SELECT WRITE FUNCTION
```

If the user wishes to write into an existing file, the file can be opened by selection of the Open function (X'04') rather than the Create function. When the file is being rewritten, the end-of-file field must be reset as follows:

```
      .
      .
      LXI  H,X'0100'                 X'0001' REVERSED
      SHLD FCBEOF                    SET TO X'0001'
      .
      .
```

In order to sequentially write the next sector each time, the following calling routine can be used:

```
    (GLBL DISK                      EXTERNAL ENTRY POINT)
    .
    .
    LXI  H,WFCB                     LOAD FCBADDR
    CALL DISK                       CALL DISK ROUTINE
    ORA  A                          CHECK RETURN CODE
    JNZ  ERROR                      HANDLE ERROR CONDITION
    .
    .
```

After writing is completed, the file must be closed, as shown below.

```
    (GLBL DISK                      EXTERNAL ENTRY POINT)
    .
    .
    LXI  H,WFCB                     ADDR OF FCB
    MVI  M,X'08'                    SELECT CLOSE
    CALL DISK                       CALL DISK
    ORA  A                          CHECK RETURN CODE
    JNZ  ERROR                      HANDLE ERROR CONDITION
    .
    .
```

The FCB for writing is initialized by the following:

```
WFCB DC  0                          FILLED IN DURING EXECUTION
     DC  X'16'                      SELECT WRITE AND CHECK
     DC  B(OBUF)                    ADDR OF WRITE BUFFER
     DC  B(128)                     LENGTH OF WRITE BUFFER
     DC  B(0)                       SECTOR # SET BY OPEN
     DC  1                          WRITE TO UNIT 1
     DC  0                          SET BY OPEN
     DC  'WRITEFILE '               FILE NAME
     DC  0                          SET BY OPEN
     DC  0                          ATTRIBUTES
```

```
FCBEOF  DC      A(0)            EOF
        DC      0               INIT ALLOCATION
        DC      0               EXTENSION SET BY OPEN
OBUF    DS      128             WRITE BUFFER
```

EIA INPUT/OUTPUT

The EIA initialization and send/receive routines allow sending/receiving of serial data via the two EIA RS-232 ports. Data are passed to and received from the EIA routines one character (byte) at a time. Parameters, which are passed to the routines in the form of a 4-byte control block, specify word length, transmission rate, and parity (odd, even or none). The address of the control block must be loaded into the HL register/Index register prior to calling the routines. All registers and flags are saved and restored by the EIA routines.

Definitions. The EIA input/output routines and control block parameters are described in the following paragraphs.

EIASET. The EIA Control Block (ECB) contains four fields representing four sequential bytes in memory (see Table 2-6). These fields provide the EIASET routine with the information necessary to initialize a port.

EIAIN. EIAIN reads one byte from the selected port (using the ECBUNIT parameter) into the accumulator. If the byte is correctly received, the carry status bit is set to zero before returning to the calling program. If an error is detected, the carry bit is set to 1.

EIAOUT. EIAOUT sends the byte passed in the accumulator to the selected port (using the ECBUNIT parameter). If the low-order bit of the ECBFLAG field is one, a break signal (200 ms of spacing level) is generated and character transmission does not take place.

The first field of the ECB, ECBFLAG, contains option bits. ECBFLAG's low-order bit is ignored by EIASET. ECBFLAG bit 1 selects two stop bits. (Normally, when bit 1 is zero, only one stop bit is transmitted. Two stop bits are required when sending data to a Teletype. In other cases, two stop bits are allowed, but are not required.) When set, bit 2 of ECBFLAG inhibits generation and checking of the parity bit. When bit 2 is zero, parity is generated and checked. Bit 3 is meaningful only when parity is being used; it specifies odd parity when set to one and even parity when set to zero.

ECBRATE. The ECBRATE field of the ECB contains a code which selects the transmission and reception (Baud) rate. Sixteen rate codes are allowed as shown in Table 2-7. Note: Using rate codes not listed in the table produces unpredictable effects.

ECBLEN. The ECBLEN field specifies the number of bits in the data to be transmitted and received. Usual values are 5, 6, 7, or 8; other character lengths cause unpredictable results. Transmission of the byte passed in the accumulator begins with the low-order bit; e.g., when 7 is specified as the length, the low-order 7 bits are transmitted.

ECBUNIT. The ECBUNIT field selects which EIA port is to be initialized. A value of one selects EIA port 1; a value of two selects EIA port 2. The EIA routines assume that port 1 is configured to simulate a terminal and port 2, a modem. After initialization by EIASET, port 1 holds its Request to Send and Data Terminal Ready lines high (see Section 3 for more information).

Table 2-6. EIA Send-Control Block

Field	# of Bytes	Functions
ECBFLAG	1	Contains flag bits having the following meaning: Bit — Function 0 — Send break. Used by EIAOUT. 1 — Selects two stop bits for TTY. 2 — Disables parity bit generation/checking. 3 — Selects odd parity. 4-7 — Unused.
ECBRATE	1	Transmission/reception rate select code (see Table 2-7).
ECBLEN	1	Length (in bits) of characters to be transmitted/received.
ECBUNIT	1	EIA line Select (1 or 2).

Table 2-7. EIA Baud Rate Select Codes

Code	Rate (Bits/Second)
0	50
1	75
2	110
3	134.5
4	150
5	300
6	600
7	600
8	1200
9	1800
A	2400
B	3600
C	4800
D	7200
E	9600
F	19200

Examples.

ASCII Data Configuration. The most common application for the EIA send/receive routines is ASCII character data exchange. ASCII data have the following specifications:

Parity	Yes/Even
Stop Bits	Two for TTY, one otherwise
Length	7 bits

ECB Communication. The EIA control block (ECB) configuration for communication with a Teletype on EIA line 2 is as follows:

```
SCBFLAG =  X'02'
SCBRATE = 2
SCBLEN = 7
SCBUNIT = 2
```

The EIA control block configuration for communication with a modem at 300 Baud on EIA line 1 is as follows:

```
SCBFLAG = X'00'
SCBRATE = 5
SCBLEN = 7
SCBUNIT = 1
```

If it is necessary to control the Request to Send or Data Terminal Ready bits or to read the Clear to Send or Data Carrier Detect status on port 1, direct I/O programming is used. For further information, refer to Section 3.

CASSETTE TAPE INPUT/OUTPUT

The cassette I/O service routines provide for reading, writing, and rewinding of cassette tapes. Data on cassette are formatted as fixed-length records separated by inter-record gaps. The gaps are sufficiently long to allow the tape to be stopped at the end of a record and restarted to read the next record. The tape I/O routines read or write a record at a time and automatically turn the tape units on and off.

The cassette tapes begin with a short length of plastic leader which must be bypassed before reading or writing begins. A flag bit indicating the first Read or Write causes the I/O routines to automatically space past the leader.

Parameters are passed to the I/O routines in the form of a 7-byte write-control block (WCB) and a 6-byte read-control block (RCB). The control block address must be loaded into the HL register pair/Index register prior to calling the Read or Write subroutines.

Error Recovery. For increased reliability, the I/O routines provide for redundant recording of data records. Any number of copies of each record can be specified and will automatically be written by the cassette Write routine. During reading, if an error is detected, the Read routine tries to read the record again. If another copy is available, it is possible to recover from the error. If the Read routine cannot find another copy, it returns with an error code.

The error recovery scheme uses sequence numbers which are recorded with the data of each record. Thus, when an error occurs, subsequent data can be automatically identified as another copy of the desired record. If the tape is manually backed up for re-try, the sequence numbers are again used to identify the desired record.

WRITE. The WCB contains the fields shown in Table 2-8. These fields represent seven sequential bytes in memory and provide the cassette Write routine with the information necessary to process the request. For the first Write, bit 0 (low-order) of WCBFLAG is set to one so that a leader is

written, and automatically reset to zero afterward. Bits 1 and 2 of WCBFLAG are used in those cases where sequences of records are to be written without inter-record gaps. When bit 1 is set to one, the Write routine leaves the cassette unit on. (Normally, the unit is turned on before writing a record and off afterwards). In conjunction with Bit 2, which suppresses the writing of an inter-record gap, multiple records can be formatted without gaps. The gap in front of the data is usually necessary to allow the tape unit sufficient time to come up to speed.

The WCBADR field of the WCB specifies that the beginning data block address be written to tape. The length of this block is specified by WCBLEN. Both WCBADR and WCBLEN are 2-byte (16-bit) fields with the low-order byte stored before the high-order (8080/85A/Z80) or high-low (6800/02) byte. The WCBUNIT field specifies the tape unit to be addressed, either 1 or 2. Finally, the WCBCOPY field specifies the number of record copies to be written. For maximum data density without automatic error recovery, one copy may be specified. For most other applications, two copies (providing a hard error rate of about one in 1xE12 bits) are adequate. All registers and flags in 8080/85A/Z80 systems are saved and restored by the cassette write routine. The 6800/02 version does not save the flags.

Table 2-8. Write Control Block

Field	Bytes	Function
WCBFLAG	1	Bits in this field request special functions as follows: Bit — Function 0 — Flags Write of first record. 1 — Leaves tape unit on after Write. 2 — Suppresses writing an inter-record gap ahead of data. 3-7 — Unused; should be zero.

Table 2-8. Write Control Block (Concluded)

Field	Bytes	Function
WCBADR	2	Address of data block to be written. Address is stored in low-order byte format followed by high-order byte (8080/85A/Z80), or high-low format (6800/02).
WCBLEN	2	Length of data block to be written. Length is stored as low-order byte, followed by high-order byte (8080/85A/Z80), or high-low (6800/02).
WCBUNIT	1	Write unit (1 or 2).
WCBCOPY	1	Number of copies of data to be written.

READ. The read-control block contains the fields shown in Table 2-9. These fields represent six sequential bytes in memory and provide the Read routine with the information necessary to process the request. As with the Write routine, the control block address must be loaded into the HL register/Index register prior to calling. For the first tape read, bit 0 of RCBFLAG is set to one. This causes the Read routine to reset the record sequence number counter to zero in order to begin reading a new tape. The bit is automatically reset to zero after the first read. Bit 1 of RCBFLAG specifies that the cassette unit is to remain on after the Read, allowing Reads where inter-record gaps are not written.

Normally, the Read routine expects the records to have sequential record numbers beginning with zero. However, under some conditions, it is useful to suppress sequential number checking and to reset the sequence number counter to the number of the next record read. Bit 2 of RCBFLAG performs this function. Once a record is read and the sequence counter is reset, zeroing bit 2 allows normal reading of sequential records.

Table 2-9. Read-Control Block

Field	Bytes	Function
RCBFLAG	1	Bits of this field request special functions as follows: Bit — Function 0 — Flags Read of first record. 1 — Leaves tape unit on after Read. 2 — Accepts record with any sequence number. 3-7 — Unused; should be 0.
RCBADR	2	Address of data block into which data from tape is to be placed. Address is stored as low-order byte, followed by high-order byte (8080/85A/Z80) or high-low (6800/02).
RCBLEN	2	Length of data block. Length is stored as low-order byte followed by high-order byte (8080/85A/Z80) or high-low (6800/02).
RCBUNIT	1	Read unit (1 or 2).

The RCBADR and RCBLEN fields specify the address and length of the read memory area. Both fields are two bytes (16 bits) long, and values are stored with the low-order byte first. RCBUNIT specifies the read unit; either 1 or 2.

Upon return from the Read routine, the accumulator contains a completion code: zero for success, non-zero for error. Several types of errors are classified by the value of the return code as follows:

Table 2-10. Tape Read Routine Error Codes

Code	Definition
1	Checksum error, normal re-try should be attempted.
2	Data on tape are not recognizable, and re-try probably will not be successful. This code results when the tape data have a different record size than specified by the RCB.

Table 2-10. Tape Read Routine Error Codes (Concluded)

Code	Definition
3	No data have been read from the tape for an unusual length of time. Possibly, a blank tape is being read.

All registers except the accumulator and the flag bits are saved and restored by the cassette Read routine. For the 6800/02, the Index register is saved.

READR. If a type "1" error is detected, recovery should be attempted as follows. After the operator manually rewinds the tape for several seconds, the cassette Read re-try routine should be called. This call requires no parameters. It causes the Read routine to search for the record by sequence number, and then attempts to reread the data. The re-try routine also returns a completion code. An error detected after re-try probably indicates an unrecoverable error. The bad record can be skipped by calling the regular Read routine again.

The re-try routine saves and restores all registers except the accumulator and the flag bits in 8080/85A/Z80 systems. The 6800/02 saves only the Index register.

Cassette Rewind Routine. When called, the Rewind routine turns on the selected tape for rewinding and waits for operator acknowledgement of completion (pressing the RETURN key). When this signal is received, the unit turns off.

Rewind requires the address of an RCB or WCB in the HL register pair/ Index register. The control block specifies the tape unit to be rewound. The routine also sets bit 0 of the control block to prepare for reading or writing of a new tape. The assumption is made that since the current tape has been rewound, a new tape will follow.

For 8080/85A/Z80 systems all registers and flags are saved and restored by the rewind routine. Flags are not saved for 6800/02 systems.

Section 3
Direct I/O Programming

The I/O service routines in KIO.R, KEYIN.R, LINEIN.R, DIO.R, EIA.R, and TIO.R provide high-level interfaces for the standard peripheral devices. However, for some applications, it may be necessary to access these devices directly. Section 3 contains the information needed to perform this task. The I/O addresses shown on the following pages apply to 8080/85A/Z80 systems. 6800/02 addresses must be mapped into memory.

All 6800/02 I/O addresses are located between X'D000' and X'DFFF'; e.g., the equivalent of an 8080/85A/Z80 IN X'F0' command is LDAA X'D0F0'. Similarly, an OUT X'F3' command is translated to STAA X'D0F3' for 6800/02 systems. All of the 6800/02 memory reference instructions may be used with the I/O devices. However, output-like commands (CLR for example) may cause problems. Commands of this type generate a read cycle at the designated I/O address. In some cases, the read I/O commands are used as reset signals. For example, the X'F2' and IN X'F3' commands are used as status latch reset pulses; thus, a CLR command at the corresponding output addresses would reset the status latches.

KEYBOARD INPUT

A status bit, interrupt mask bit and four I/O commands are associated with the keyboard. The I/O commands are summarized in Table 3-1.

Table 3-1. Keyboard Input Commands

I/O CMD	Function	I/O Bit Assignments							
		7	6	5	4	3	2	1	0
IN X'F0'	Read data	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
IN X'F1'	Read status	-	-	-	-	-	jmpr	brk	stat
IN X'F2'	Reset status	-	-	-	-	-	-	-	-
OUT X'F1'	Set mask	-	-	-	-	act led	kbd led	brk mask	key mask

When a key is depressed, the keyboard status bit is set to one. The IN X'F1' command reads the status and moves it to the low-order bit of the accumulator, where it can be read by the IN X'F0' command. Keyboard data are valid only during the time that keys are depressed; therefore, data must be read within approximately 100 ms from the time the status bit is set to one. The status bit remains set until reset by the IN X'F2' command. The low-order seven bits of the data byte from the keyboard contain the ASCII encoding of the character entered.

The interrupt mask bit can be used to enable interrupts from the keyboard status bit. When the master system reset is generated at power-on or by the RESET key, the mask bit is cleared and keyboard interrupts are disabled. The keyboard interrupts are enabled when the "set mask" command (OUT X'F1') outputs an X'01'. CPU interrupts must be enabled by an "EI" instruction (CLI for 6800/02) before they can be accepted.

The interrupt causes a CALL to location X'0028' for 8080/85A/Z80 systems or a JSR to location X'FFF8' for 6800/02 systems. CPU interrupt processing results in interrupt disablement. The "reset status" command (IN X'F2') must, therefore, be executed before the next EI instruction. Keyboard interrupts are also disabled when the "set mask" command outputs an X'00'.

BREAK KEY

The BREAK key is provided as an interrupt-generating escape key and is used by the Debugger for this function. Note: the status and mask bits are separate from those of other keyboard characters. A status bit, an interrupt mask, and three I/O commands are provided for the BREAK key as summarized in Table 3-2.

Table 3-2. Break Key Commands

I/O CMD	Function	I/O Bit Assignments							
		7	6	5	4	3	2	1	0
IN X'F1'	Read status	-	-	-	-	-	-	stat	-
IN X'F3'	Reset status	-	-	-	-	-	-	-	-
OUT X'F1'	Set mask	-	-	-	-	-	-	mask	-

When the BREAK key is depressed, the break status bit is set to one. The IN X'F1' command moves the status bit into bit 1 of the accumulator. The IN X'F3' command resets the status bit.

The mask bit can be used to enable interrupts from the break status bit. BREAK key interrupts are disabled when the master system Reset clears the break mask bit, or when the OUT X'F1' command outputs an X'00'. When the OUT X'F1' command sets the mask bit to one, BREAK key interrupts are enabled. An "EI" instruction enables the CPU for recognition of the interrupts in 8080 systems. A "SIM" instruction enables RST 5.5 in the 8085A and allows the interrupt to be recognized. In 6800, 6802 and Z80 systems, BREAK key interrupts are vectored through the Non-Maskable Interrupt; thus, the "SEI" and "CLI" or "EI" and "DI" instructions have no effect.

The BREAK key interrupt causes a CALL (or JSR) to location X'0020' in 8080 systems, X'FFFC' (Non-Maskable Interrupt) in 6800 systems, X'66' in Z80 systems and X'2C' in 8085A systems. The interrupt remains set at these locations, pending a reset of the status bit.

CRT DISPLAY SELECT

The CRT display interface is able to refresh the display from any 2K page in memory. The display page is selected as shown below in Table 3-3.

Table 3-3. CRT Display Select

I/O CMD	Function	I/O Bit Assignments							
		7	6	5	4	3	2	1	0
OUT X'F0'	Addr slct	Addr 15	Addr 14	Addr 13	Addr 12	Addr 11	Addr 10	-	-

CRT display address selection is accomplished by loading the accumulator with the high-order byte of the display page address and executing the OUT X'F0' command.

Note that the CRT display must be refreshed from a page of memory beginning on an integral 2K boundary, rather than an arbitrary 2K memory block.

CRT DISPLAY OUTPUT

Since the CRT display is refreshed from memory, direct output is accomplished by storing data in the memory page selected for refresh. Data are converted by the CRT display module into a video signal which represents each ASCII code as a 9 x 7 dot matrix. Since only 128 displayable characters are generated, only the low-order seven bits of each byte are required for the display. The high order bit is used to select display field attributes as shown in Table 3-4.

Table 3-4. CRT Field Attributes

Display Code	Function
X'80'	Reset display to normal mode
X'81'	Highlight display
X'82'	Blink display
X'90'	Reverse video display
X'83'	Blink and highlight
X'91'	Reverse and highlight
X'92'	Reverse and blink
X'93'	Reverse, blink and highlight

Each field attribute character occupies one memory location, but does not occupy any of the display locations. Thus, the CRT display buffer size is fixed only if the number of display attribute codes is fixed.

When there are no field attribute codes, the display is formatted from the CRT refresh memory as follows. The character at the top left of the screen is generated from the first byte. Subsequent bytes correspond to sequential characters across the 80-character line, with the next line using the next 80 characters, etc. Table 3-5 shows the offsets from the beginning of the display page, corresponding to the first character of each line (assuming there are no field attribute characters). As noted above, the CRT display can be selected to refresh from any 2K memory page.

AMDS system routines set the display page to the last 2K page in memory (X'D800'). For the 6800/02 the corresponding page begins at X'F800'.

Table 3-5. CRT Display Line Offsets

Line	Offset	Line	Offset	Line	Offset
1	+0	9	+640	17	+1280
2	+80	10	+720	18	+1360
3	+160	11	+800	19	+1440
4	+240	12	+880	20	+1520
5	+320	13	+960	21	+1600
6	+400	14	+1040	22	+1680
7	+480	15	+1120	23	+1760
8	+560	16	+1200	24	+1840

MEMORY PROTECTION

Each 8K memory block contains an 8-bit protection register which can be read and reset by I/O commands. Each register bit indicates the write-protect status of a 1K page of the block. The protection I/O commands are summarized in Tables 3-6 and 3-7.

Table 3-6. Protect Command Bit Assignments

7	6	5	4	3	2	1	0
Page 7	Page 6	Page 5	Page 4	Page 3	Page 2	Page 1	Page 0

Table 3-7. Memory Protect Commands

I/O CMD	Function
IN X'1F'	read module 0 protection (X'0000'-X'1FFF')
IN X'3F'	read module 1 protection (X'2000'-X'3FFF')
IN X'5F'	read module 2 protection (X'4000'-X'5FFF')
IN X'7F'	read module 3 protection (X'6000'-X'7FFF')
IN X'9F'	read module 4 protection (X'8000'-X'9FFF')
IN X'BF'	read module 5 protection (X'A000'-X'BFFF')
IN X'DF'	read module 6 protection (X'C000'-X'DFFF')
	6800 read module 6 protection (X'E000'-X'FFFF')
OUT X'1F'	set module 0 protection (X'0000'-X'1FFF')
OUT X'3F'	set module 1 protection (X'2000'-X'3FFF')
OUT X'5F'	set module 2 protection (X'4000'-X'5FFF')
OUT X'7F'	set module 3 protection (X'6000'-X'7FFF')
OUT X'9F'	set module 4 protection (X'8000'-X'9FFF')
OUT X'BF'	set module 5 protection (X'A000'-X'BFFF')
OUT X'DF'	set module 6 protection (X'C000'-X'DFFF')
	6800 set module 6 protection (X'E000'-X'FFFF')

The low-order bit of the protection register is assigned to the first 1K page of the block, the high-order bit is assigned to the last 1K page, and the other bits are assigned to respective 1K pages. A bit set to one causes a write request to be inhibited at any address in the corresponding 1K page. A bit set to zero allows writing into memory. Execution of the boot PROM program clears all protection registers and leaves memory unprotected.

Each memory block has one input command to read the protection register and one output command to reset the protection register. These commands along with the range of memory addresses controlled by each I/O command are listed in Table 3-7. Note: Module 7 of the 8080 processor is assigned to the bootstrap loader PROM in the AMDS and cannot be implemented.

REAL-TIME CLOCK

The Multipurpose I/O card (MPIO) contains a three-channel real-time clock. Two channels are dedicated to the high speed serial data link and are not directly available to the user. The third channel is available to the user as a programmable clock or interval timer. The I/O ports associated with the real-time clock are shown in Table 3-8.

Table 3-8. Real-Time Clock Commands

Function	Command	Bit(s)	Bit Function
READ RTC	IN X'CA'	0-7	COUNTER LSB'S
READ RTC	IN X'CA'	0-7	COUNTER MSB'S
LOAD RX CLOCK	OUT X'C8'	0-7	COUNTER LSB'S
LOAD RX CLOCK	OUT X'C8'	0-7	COUNTER MSB'S
LOAD TX CLOCK	OUT X'C9'	0-7	COUNTER LSB'S
LOAD TX CLOCK	OUT X'C9'	0-7	COUNTER MSB'S
LOAD RTC	OUT X'CA'	0-7	COUNTER LSB'S
LOAD RTC	OUT X'CA'	0-7	COUNTER MSB'S
LOAD CTL WORD	OUT X'CB'	0	1 = BCD, 0 = HEX
		1-3	COUNTER MODE
		4,5	MSB, LSB SELECT
		6,7	COUNTER SELECT
TRIGGER RTC	IN X'CE'	-	NO FUNCTION

For additional information on the real-time clock, see the 8253 "Programmable Interval Timer" specifications in the Appendix.

SERIAL I/O PORTS

The MPIO card includes two serial I/O ports which are implemented by LSI USART's. There are four input and four output addresses associated with each port. Port 1 uses X'C1' through X'C3', and port 2 uses X'C4' through X'C7'. The functions for both ports are listed in Tables 3-9 through 3-12. For more information on serial I/O port programming, refer to the 2651 "Programmable Communications Interface" specifications in the appendix.

Table 3-9. Serial Port 1 Read Commands

Function	Command	Bit(s)	Bit Function
READ DATA	IN X'C0'	0-7	INPUT DATA
READ STAT REG SYNC	IN X'C1'	0	TX REG EMPTY
		1	RX REG EMPTY
		2	CHANGE IN DCD OR DSR
		3	PARITY ERROR OR DLE
		4	OVERRUN
		5	SYNC DETECT
		6	DCD HIGH
		7	DSR HIGH
READ STAT REG ASYNC	IN X'C1'	0	TX REG EMPTY
		1	RX REG EMPTY
		2	CHANGE IN DCD OR DSR
		3	PARITY ERROR
		4	OVERRUN
		5	FRAMING ERROR
		6	DCD HIGH
		7	DSR HIGH
READ MODE REG SYNC(REG1)	IN X'C2'	0-1	MODE AND BAUD RATE MPLY
		2-3	CHARACTER LENGTH
		4	PARITY ENABLE
		5	PARITY EVEN
		6	1 = TRANSPARENCY MODE
		7	1 = SINGLE SYNC

Table 3-9. Serial Port 1 Read Commands (Concluded)

Function	Command	Bit(s)	Bit Function
READ MODE REG ASYNC(REG1)	IN X'C2'	0-1	MODE AND BAUD RATE MPLY
		2-3	CHARACTER LENGTH
		4	PARITY ENABLE
		5	PARITY EVEN
		6-7	NUMBER STOP BITS
READ MODE REG (REG2)	IN X'C2'	0-3	BAUD RATE SELECTION
		4-5	MUST BE 1
		6-7	NO FUNCTION
READ CMD REG SYNC	IN X'C3'	0	TX ENABLE
		1	FORCE DTR HIGH
		2	RX ENABLE
		3	SEND DLE
		4	RESET ERROR CONDITION
		5	FORCE RTS HIGH
		6-7	OPERATING MODE
READ CMD REG ASYNC	IN X'C3'	0	RX ENABLE
		1	FORCE DTR HIGH
		2	RX ENABLE
		3	SEND BREAK
		4	RESET ERROR CONDITION
		5	FORCE RTS HIGH
		6-7	OPERATING MODE

Table 3-10. Serial Port 1 Write Commands

Function	Command	Bit(s)	Bit Function
WRITE DATA	OUT X'C0'	0-7	TRANSMIT DATA
WRITE SYN1	OUT X'C1'	0-7	SYNC CHARACTER 1
WRITE SYN2	OUT X'C1'	0-7	SYNC CHARACTER 2
WRITE DLE	OUT X'C1'	0-7	DATA LINK ESCAPE CHAR
WRITE MODE REG1(SYNC)	OUT X'C2'	0-1	MODE AND BAUD RATE MPLY
		2-3	CHARACTER LENGTH
		4	PARITY ENABLE
		5	PARITY EVEN
		6	1 = TRANSPARENCY MODE
		7	1 = SINGLE SYNC CHAR.
WRITE MODE REG1(ASYNC)	OUT X'C2'	0-1	MODE AND BAUD RATE MPLY
		2-3	CHARACTER LENGTH
		4	PARITY ENABLE
		5	PARITY EVEN
		6-7	NUMBER STOP BITS
WRITE MODE REG2	OUT X'C2'	0-3	BAUD RATE SELECT
		4-5	MUST BE 1
		6-7	DON'T CARE
WRITE CMD REG SYNC	OUT X'C3'	0	TX ENABLE
		1	FORCE DTR HIGH
		2	RX ENABLE
		3	SEND DLE
		4	RESET ERROR
		5	FORCE RTS HIGH
		6-7	SELECT OPER MODE
WRITE CMD REG ASYNC	OUT X'C3'	0	TX ENABLE
		1	FORCE DTR HIGH
		2	RX ENABLE
		3	SEND BREAK
		4	RESET ERROR
		5	FORCE RTS HIGH
		6-7	SELECT OPER MODE

Table 3-11. Serial Port 2 Read Commands

Function	Command	Bit(s)	Bit Function
READ DATA	IN X'C4'	0-7	INPUT DATA
READ STAT REG	IN X'C5'	0	TX REG EMPTY
SYNC		1	RX REG EMPTY
		2	CHANGE IN DCD OR DSR
		3	PARITY ERROR OR DLE
		4	OVERRUN
		5	SYNC DETECT
		6-7	NO FUNCTION
READ STAT REG	IN X'C5'	0	RX REG EMPTY
ASYNC		1	RX REG EMPTY
		2	CHANGE IN DCD OR DSR
		3	PARITY ERROR
		4	OVERRUN
		5	FRAMING ERROR
		6-7	NO FUNCTION
READ MODE REG	IN X'C6'	0-1	MODE AND BAUD RATE MPLY
SYNC(REG1)		2-3	CHARACTER LENGTH
		4	PARITY ENABLE
		5	PARITY EVEN
		6	1 = TRANSPARENCY MODE
		7	1 = SINGLE SYNC
READ MODE REG	IN X'C6'	0-1	MODE AND BAUD RATE MPLY
ASUNC(REG1)		2-3	CHARACTER LENGTH
		4	PARITY ENABLE
		5	PARITY EVEN
		6-7	NUMBER STOP BITS
READ MODE REG	IN X'C6'	0-3	BAUD RATE SELECTION
		4	1 = INTERNAL RX CLOCK
		5	1 = INTERNAL TX CLOCK
		6-7	NO FUNCTION

Table 3-11. Serial Port 2 Read Commands (Concluded)

Function	Command	Bit(s)	Bit Function
READ CMD REG SYNC	IN X'C7'	0	TX ENABLE
		1	FORCE DTR LOW
		2	RX ENABLE
		3	SEND DLE
		4	RESET ERROR CONDITION
		5	NO FUNCTION
		6-7	OPERATING MODE
READ CMD REG ASYNC	IN X'C7'	0	TX ENABLE
		1	FORCE DTR LOW
		2	RX ENABLE
		3	SEND BREAK
		4	RESET ERROR CONDITION
		5	NO FUNCTION
		6-7	OPERATING MODE

Table 3-12. Serial Port 2 Write Commands

Function	Command	Bit(s)	Bit Function
WRITE DATA	OUT X'C4'	0-7	TRANSMIT DATA
WRITE SYN1	OUT X'C5'	0-7	SYNC CHARACTER 1
WRITE SYN2	OUT X'C5'	0-7	SYNC CHARACTER 2
WRITE DLE	OUT X'C5'	0-7	DATA LINK ESCAPE CHAR.
WRITE MODE REG1(SYNC)	OUT X'C6'	0-1	MODE AND BAUD RATE MPLY
		2-3	CHARACTER LENGTH
		4	PARITY ENABLE
		5	PARITY EVEN
		6	1 = TRANSPARENCY MODE
		7	1 = SINGLE SYNC CHAR
WRITE MODE REG1(ASYNC)	OUT X'C6'	0-1	MODE AND BAUD RATE MPLY
		2-3	CHARACTER LENGTH
		4	PARITY ENABLE
		5	PARITY EVEN
		6-7	NUMBER STOP BITS

Table 3-12. Serial Port 2 Write Commands (Concluded)

Function	Command	Bit(s)	Bit Function
WRITE MODE REG2	OUT X'C6'	0-3	BAUD RATE SELECT
		4-5	MUST BE 1
		6-7	DON'T CARE
WRITE CMD REG SYNC	OUT X'C7'	0	TX ENABLE
		1	NO FUNCTION
		2	RX ENABLE
		3	SEND DLE
		4	RESET ERROR
		5	NO FUNCTION
		6-7	SELECT OPER MODE
WRITE CMD REG ASYNC	OUT X'C7'	0	TX ENABLE
		1	NO FUNCTION
		2	RX ENABLE
		3	SEND BREAK
		4	RESET ERROR
		5	NO FUNCTION
		6-7	SELECT OPER MODE

DISK I/O PORT

The disk I/O port is parallel, bi-directional, and capable of transmitting and receiving data. Write data are guaranteed stable at the trailing (rising) edge of WRST-. Neither write nor read data are latched. Table 3-13 lists the I/O commands associated with the disk I/O port.

Table 3-13. Disk I/O Port Commands

Function	Command	Bit(s)	Bit Function
READ STATUS	IN X'D8'	0-7	INPUT STATUS
READ DATA	IN X'D9'	0-7	INPUT DATA
WRITE COMMAND	OUT X'D8'	0-7	OUTPUT COMMAND
WRITE DATA	OUT X'D9'	0-7	OUTPUT DATA
DISK RESET	OUT X'CE'	-	NO FUNCTION

PRINTER PORT

A "Centronics"-compatible parallel interface with latched output data is provided by the printer port. Data are guaranteed stable both before and after DATA STROBE-.

The I/O commands associated with the printer are listed in Table 3-14.

Table 3-14. Printer I/O Commands

Function	Command	Bit(s)	Bit Function
WRITE DATA	OUT X'CD'	0-7	OUTPUT DATA
READ STATUS	IN X'CD'	6	PAPER OUT
		7	PRINTER BUSY

BOARD STATUS PORT

The AMDS provides an overall board status port that inputs status for the disk I/O ports, serial port 1, RTC, two user-definable pins and the printer port. The I/O command for the port is shown in Table 3-15.

Table 3-15. Board Status Port Commands

Function	Command	Bit(s)	Bit Function
READ STATUS	IN X'CD'	0	DISK SERVICE REQUEST
		1	SERIAL PORT 1 READY
		2	REAL-TIME CLOCK OVERFLOW
		3	USER DEFINED FUNCTION
		4	USER DEFINED FUNCTION
		5	RESERVED FOR DISK EXPANSION
		6	PRINTER PAPER OUT
		7	PRINTER BUSY

BOARD COMMAND PORT

The overall board command port controls interrupt masks, high-speed data link transmit enable, and disk controller select lines. The two commands associated with this port are shown below in Table 3-16.

Table 3-16. Board Command Port Commands

Function	Command	Bit(s)	Bit Function
WRITE CONTROL	OUT X'CC'	0	SELECT DISK CONTROLLER 1
		1	SELECT DISK CONTROLLER 2
		2	SELECT DISK CONTROLLER 3
		3	ENABLE NETWORK TX
		4	ENABLE SERIAL PORT 1 INTERRUPT
		5	ENABLE PRINTER INTERRUPT
		6	ENABLE RTC INTERRUPT
		7	ENABLE DISK INTERRUPT
READ CONTROL	IN X'CC'	0-7	Same as above

JUMPER SELECTION

There are 18 jumpers on the MPIO board, most of which are used to reconfigure serial port 1 as a terminal or modem. The remaining jumpers select data inputs to serial port 2, and switch board addresses. Some are mutually exclusive. The jumper functions are listed in Table 3-17.

Table 3-17. Jumper Functions

Jumper	Function
1	Connects TXD to USART1 input data line. Used when port 1 is configured as a modem. 6 must be open.
2	Connects TXD to USART1 output data line. Used when port 1 is configured as a terminal.
3	Connects USART1 RTS line to RTS. Used when port 1 is configured as a terminal.
4	Connects DTR to USART1 DTR line. Used when port 1 is configured as a terminal.
5	Connects RXD to USART1 output line. Used when port 1 is configured as a modem.
6	Connects RXD to USART1 input line. Used when port 1 is configured as a terminal. 1 must be open.
7	Connects CTS to USART1 C13 input. Used when port 1 is configured as a terminal. 10 must be open.
8	Connects DCD to USART1 DCD line. Used when port 1 is configured as a terminal. 9 must be open.
9	Connects DTR to DCD input. 8 must be open.
10	Connects RTS output to CTS input. 7 must be open.
11	Connects port 2 RS232 TXD line to USART2 input line if jumper 12 is closed. 14 must be open.
12	Connects RS-232/C TXD or current loop RXD to USART2 input. 13 must be open.
13	Connects high speed links to USART2 input. 12 must be open.
14	Connects current loop RXD to USART2 input if 12 is closed. 11 must be open.

Table 3-17. Jumper Functions (Concluded)

Jumper	Function
J5	User definable bit 1
J6	User definable bit 2
S5	Maps all I/O addresses into X'AX'.
S6	Maps all I/O addresses into X'CX'.

Table 3-18. Summary of I/O Port Addresses

Port Address 8080/85A/Z80	Port Address 6800/02	Associated Function
C0-C3	D0C0-D0C0	SERIAL PORT 1
C4-C7	D0C4-D0C7	SERIAL PORT 2
C8-C8	D0C8-D0C8	REAL-TIME CLOCK
CC	D0CC	MPIO BOARD CONTROL PORT
CD	D0CD	PRINTER PORT
CD	D0CD	MPIO BOARD STATUS PORT
CE	D0CE	DISK RESET
CE	D0CE	TRIGGER RTC
D8-D9	D0D8-D0D9	DISK
F0-F2	D0F0-D0F2	KEYBOARD
F0	D0F0	CRT PAGE SELECT
F1,F3	D0F1,D0F3	BREAK KEY
F4,F5		8080 EMULATOR CONTROL
F6*+		Z80,8085 EMULATOR CONTROL
	FFF4-FFF7*	6800,6802 EMULATOR CONTROL
F8-FE	D0F8-D0FE	DEBUG/PROM PGMR

* These registers alter emulation mode.

+ This I/O register is not available during emulation. Attempts to write into port F6 during emulation may produce unpredictable results.

8255A FUNCTIONAL DESCRIPTION

General

The 8255A is a programmable peripheral interface (PPI) device designed for use in Intel® microcomputer systems. Its function is that of a general purpose I/O component to interface peripheral equipment to the microcomputer system bus. The functional configuration of the 8255A is programmed by the system software so that normally no external logic is necessary to interface peripheral devices or structures.

Data Bus Buffer

This 3-state bidirectional 8-bit buffer is used to interface the 8255A to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU. Control words and status information are also transferred through the data bus buffer.

Read/Write and Control Logic

The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control busses and in turn, issues commands to both of the Control Groups.

($\overline{CS}$)

Chip Select. A "low" on this input pin enables the communiction between the 8255A and the CPU.

($\overline{RD}$)

Read. A "low" on this input pin enables the 8255A to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to "read from" the 8255A.

($\overline{WR}$)

Write. A "low" on this input pin enables the CPU to write data or control words into the 8255A.

(A_0 and A_1)

Port Select 0 and Port Select 1. These input signals, in conjunction with the RD and WR inputs, control the selection of one of the three ports or the control word registers. They are normally connected to the least significant bits of the address bus (A_0 and A_1).

8255A BASIC OPERATION

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	INPUT OPERATION (READ)
0	0	0	1	0	PORT A → DATA BUS
0	1	0	1	0	PORT B → DATA BUS
1	0	0	1	0	PORT C → DATA BUS
					OUTPUT OPERATION (WRITE)
0	0	1	0	0	DATA BUS → PORT A
0	1	1	0	0	DATA BUS → PORT B
1	0	1	0	0	DATA BUS → PORT C
1	1	1	0	0	DATA BUS → CONTROL
					DISABLE FUNCTION
X	X	X	X	1	DATA BUS → 3-STATE
1	1	0	1	0	ILLEGAL CONDITION
X	X	1	1	0	DATA BUS → 3-STATE

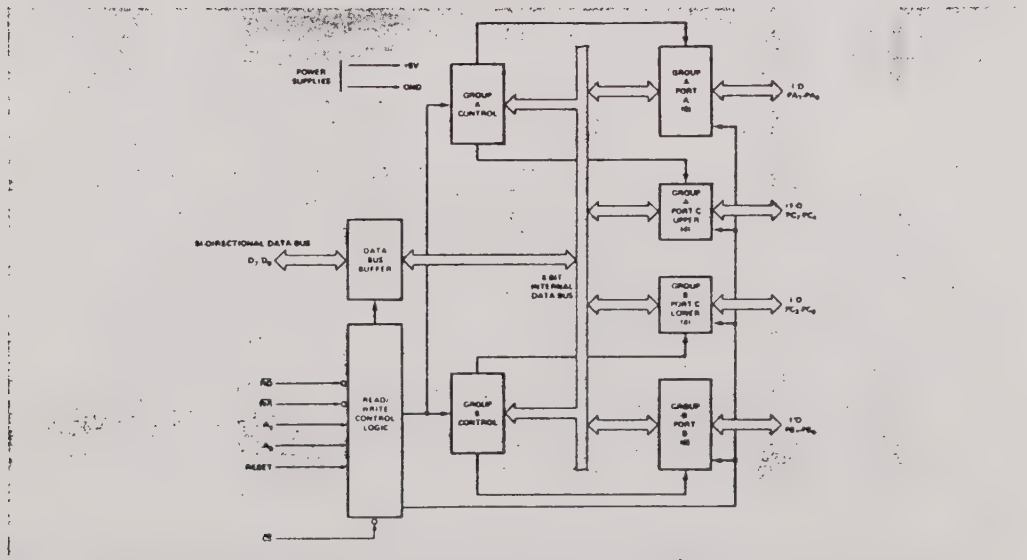


Figure 1. 8255A Block Diagram Showing Data Bus Buffer and Read/Write Control Logic Functions

00744A

(RESET)

Reset. A "high on this input clears the control register and all ports (A, C, C) are set to the input mode.

Group A and Group B Controls

The functional configuration of each port is programmed by the systems software. In essence, the CPU "outputs" a control word to the 8255A. The control word contains information such as "mode", "bit set", "bit reset", etc., that initializes the functional configuration of the 8255.

Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control Logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.

Control Group A – Port A and Port C upper (C7-C4)
Control Group B – Port B and Port C lower (C3-C0)

The Control Word Register can Only be written into. No Read operation of the Control Word Register is allowed.

Ports A, B, and C

The 8255A contains three 8-bit ports (A, B, and C). All can be configured in a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 8255A.

Port A. One 8-bit data output latch/buffer and one 8-bit data input latch.

Port B. One 8-bit data input/output latch/buffer and one 8-bit data input buffer.

Port C. One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signal inputs in conjunction with ports A and B.

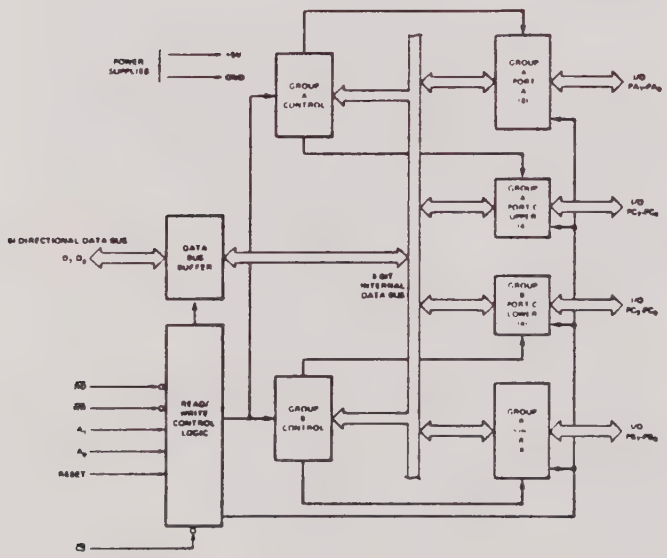
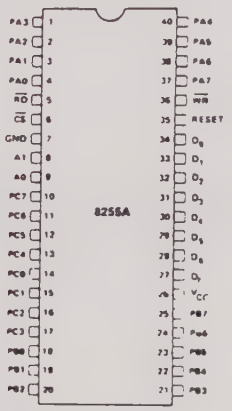


Figure 2. 8225A Block Diagram Showing Group A and Group B Control Functions

PIN CONFIGURATION



PIN NAMES

D_7-D_0	DATA BUS (BI DIRECTIONAL)
RESET	RESET INPUT
$\overline{CS}$	CHIP SELECT
$\overline{RD}$	READ INPUT
$\overline{WR}$	WRITE INPUT
A0, A1	PORT ADDRESS
PA7-PA0	PORT A (BIT)
PB7-PB0	PORT B (BIT)
PC7-PC0	PORT C (BIT)
V_{CC}	+5 VOLTS
GND	0 VOLTS

00744A

8255A OPERATIONAL DESCRIPTION

Mode Selection

There are three basic modes of operation that can be selected by the system software:

Mode 0 – Basic Input/Output
Mode 1 – Strobed Input/Output
Mode 2 – Bi-Directional Bus

When the reset input goes "high" all ports will be set to the input mode (i.e., all 24 lines will be in the high impedance state). After the reset is removed the 8255A can remain in the input mode with no additional initialization required. During the execution of the system program any of the other modes may be selected using a single output instruction. This allows a single 8255A to service a variety of peripheral devices with a simple software maintenance routine.

The modes for Port A and Port B can be separately defined, while Port C is divided into two portions as required by the Port A and Port B definitions. All of the output registers, including the status flip-flops, will be reset whenever the mode is changed. Modes may be combined so that their functional definition can be "tailored" to almost any I/O structure. For instance; Group B can be programmed in Mode 0 to monitor simple switch closings or display computational results, Group A could be programmed in Mode 1 to monitor a keyboard or tape reader on an interrupt-driven basis.

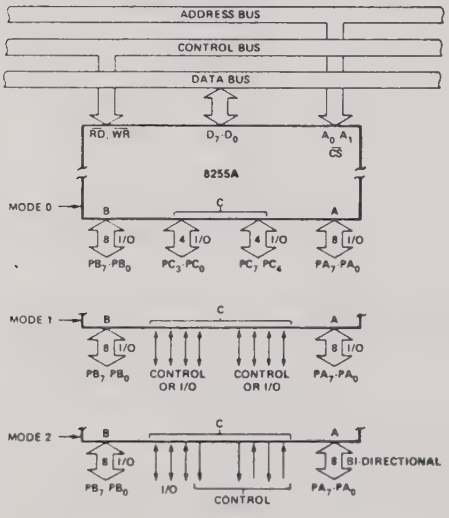


Figure 3. Basic Mode Definitions and Bus Interface

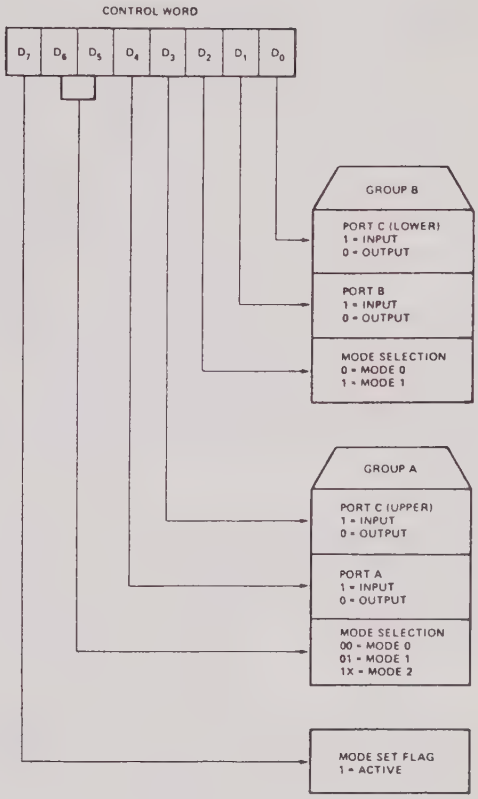


Figure 4. Mode Definition Format

The mode definitions and possible mode combinations may seem confusing at first but after a cursory review of the complete device operation a simple, logical I/O approach will surface. The design of the 8255A has taken into account things such as efficient PC board layout, control signal definition vs PC layout and complete functional flexibility to support almost any peripheral device with no external logic. Such design represents the maximum use of the available pins.

Single Bit Set/Reset Feature

Any of the eight bits of Port C can be Set or Reset using a single OUTput instruction. This feature reduces software requirements in Control-based applications.

00744A

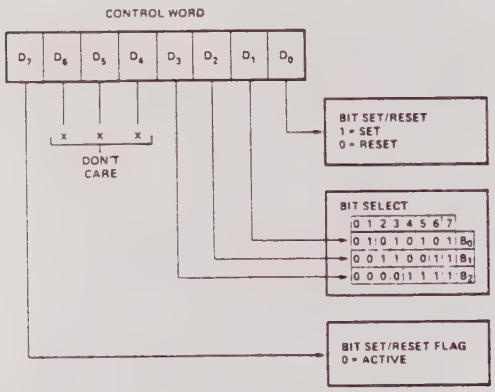

CONTROL WORD
D7 D6 D5 D4 D3 D2 D1 D0
X X X
DON'T CARE
BIT SET/RESET
1 = SET
0 = RESET
BIT SELECT
0 1 2 3 4 5 6 7
0 1 0 1 0 1 0 1 B0
0 0 1 1 0 0 1 1 B1
0 0 0 0 1 1 1 1 B2
BIT SET/RESET FLAG
0 = ACTIVE

Figure 5. Bit Set/Reset Format

When Port C is being used as status/control for Port A or B, these bits can be set or reset by using the Bit Set/Reset operation just as if they were data output ports.

Interrupt Control Functions

When the 8255A is programmed to operate in mode 1 or mode 2, control signals are provided that can be used as interrupt request inputs to the CPU. The interrupt request signals, generated from port C, can be inhibited or enabled by setting or resetting the associated INTE flip-flop, using the bit set/reset function of port C.

This function allows the Programmer to disallow or allow a specific I/O device to interrupt the CPU without affecting any other device in the interrupt structure.

INTE flip-flop definition:

(BIT-SET) — INTE is SET — Interrupt enable
(BIT-RESET) — INTE is RESET — Interrupt disable

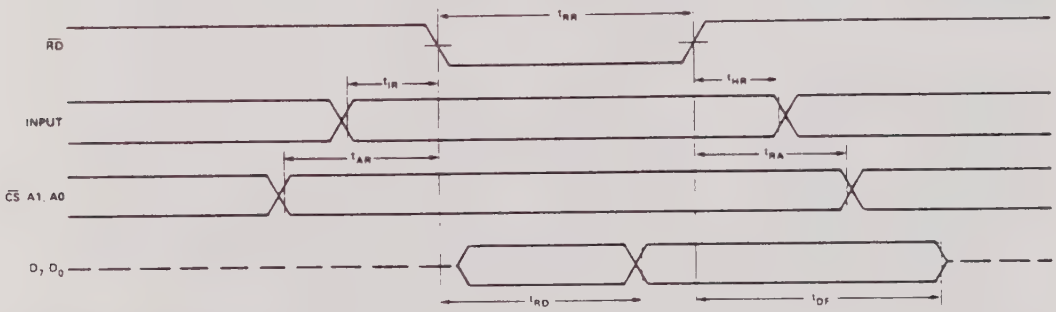
Note: All Mask flip-flops are automatically reset during mode selection and device Reset.

Operating Modes

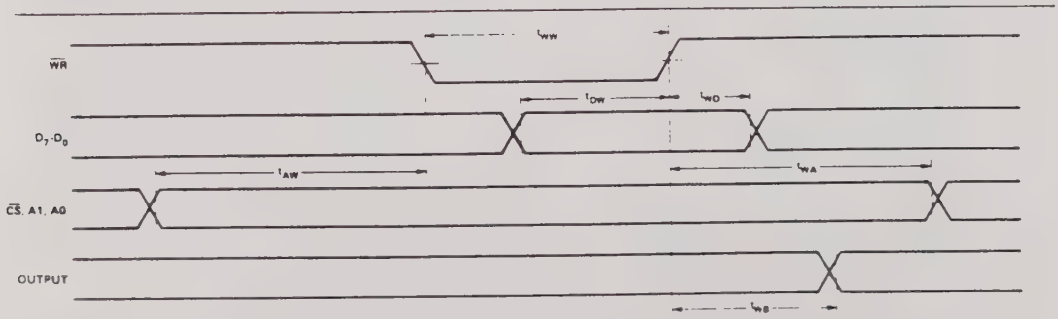
MODE 0 (Basic Input/Output). This functional configuration provides simple input and output operations for each of the three ports. No "handshaking" is required, data is simply written to or read from a specified port.

Mode 0 Basic Functional Definitions:

- Two 8-bit ports and two 4-bit ports.
- Any port can be input or output.
- Outputs are latched.
- Inputs are not latched.
- 16 different Input/Output configurations are possible in this Mode.



MODE 0 (Basic Input)

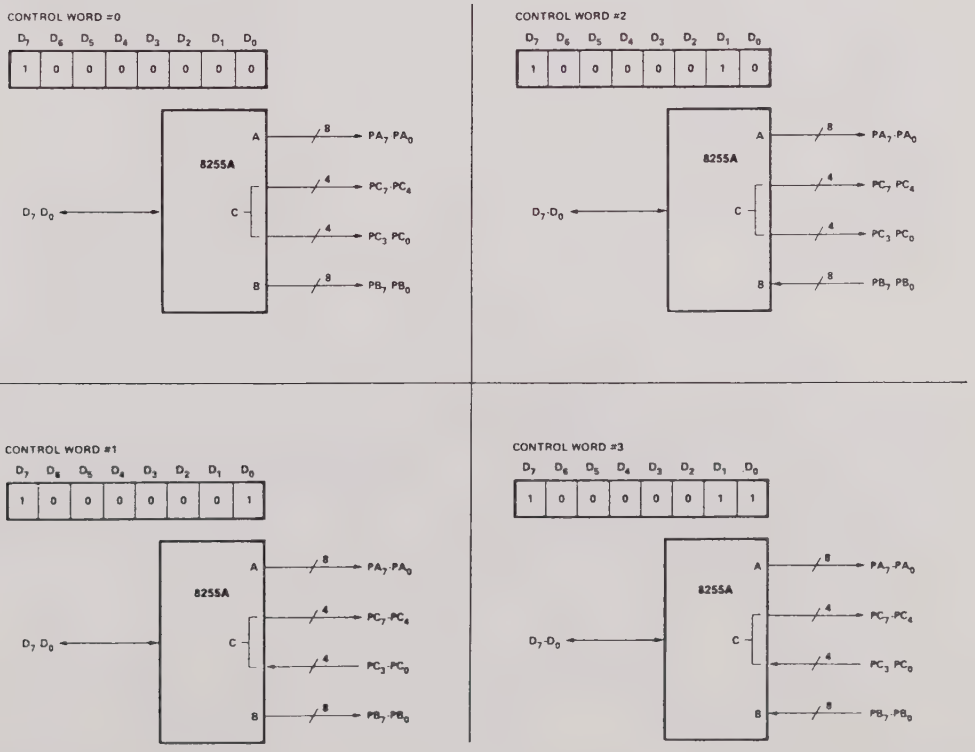


MODE 0 (Basic Output)

MODE 0 Port Definition

A		B		GROUP A			GROUP B	
D_4	D_3	D_1	D_0	PORT A	PORT C (UPPER)	#	PORT B	PORT C (LOWER)
0	0	0	0	OUTPUT	OUTPUT	0	OUTPUT	OUTPUT
0	0	0	1	OUTPUT	OUTPUT	1	OUTPUT	INPUT
0	0	1	0	OUTPUT	OUTPUT	2	INPUT	OUTPUT
0	0	1	1	OUTPUT	OUTPUT	3	INPUT	INPUT
0	1	0	0	OUTPUT	INPUT	4	OUTPUT	OUTPUT
0	1	0	1	OUTPUT	INPUT	5	OUTPUT	INPUT
0	1	1	0	OUTPUT	INPUT	6	INPUT	OUTPUT
0	1	1	1	OUTPUT	INPUT	7	INPUT	INPUT
1	0	0	0	INPUT	OUTPUT	8	OUTPUT	OUTPUT
1	0	0	1	INPUT	OUTPUT	9	OUTPUT	INPUT
1	0	1	0	INPUT	OUTPUT	10	INPUT	OUTPUT
1	0	1	1	INPUT	OUTPUT	11	INPUT	INPUT
1	1	0	0	INPUT	INPUT	12	OUTPUT	OUTPUT
1	1	0	1	INPUT	INPUT	13	OUTPUT	INPUT
1	1	1	0	INPUT	INPUT	14	INPUT	OUTPUT
1	1	1	1	INPUT	INPUT	15	INPUT	INPUT

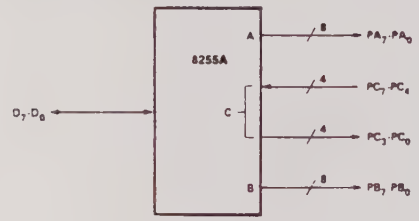
MODE 0 Configurations



00744A

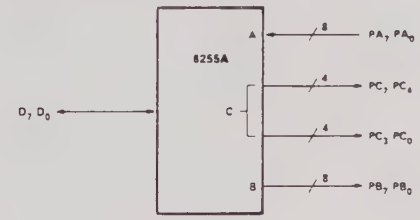
CONTROL WORD #4

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	0	0



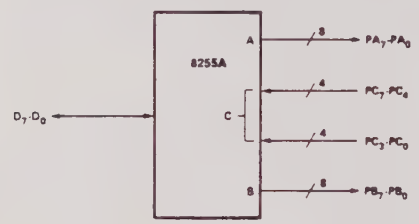
CONTROL WORD #8

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	0	0



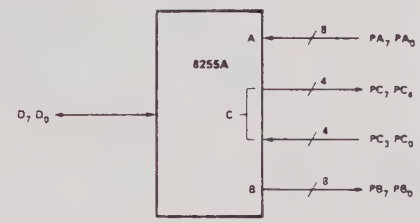
CONTROL WORD #5

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	0	1



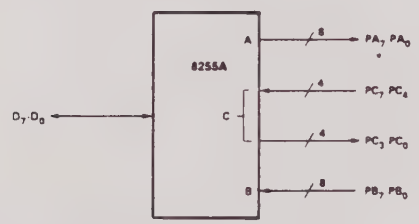
CONTROL WORD #9

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	0	1



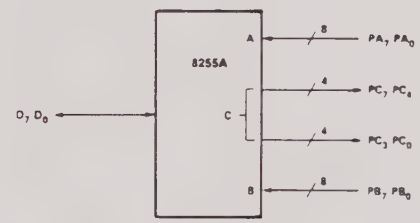
CONTROL WORD #6

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	0



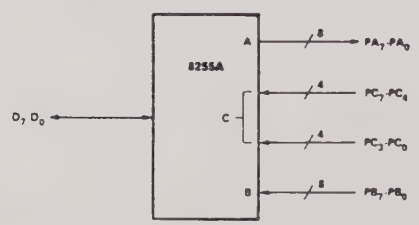
CONTROL WORD #10

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	1	0



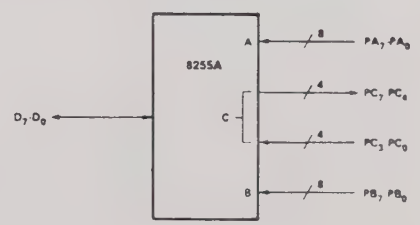
CONTROL WORD #7

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	1

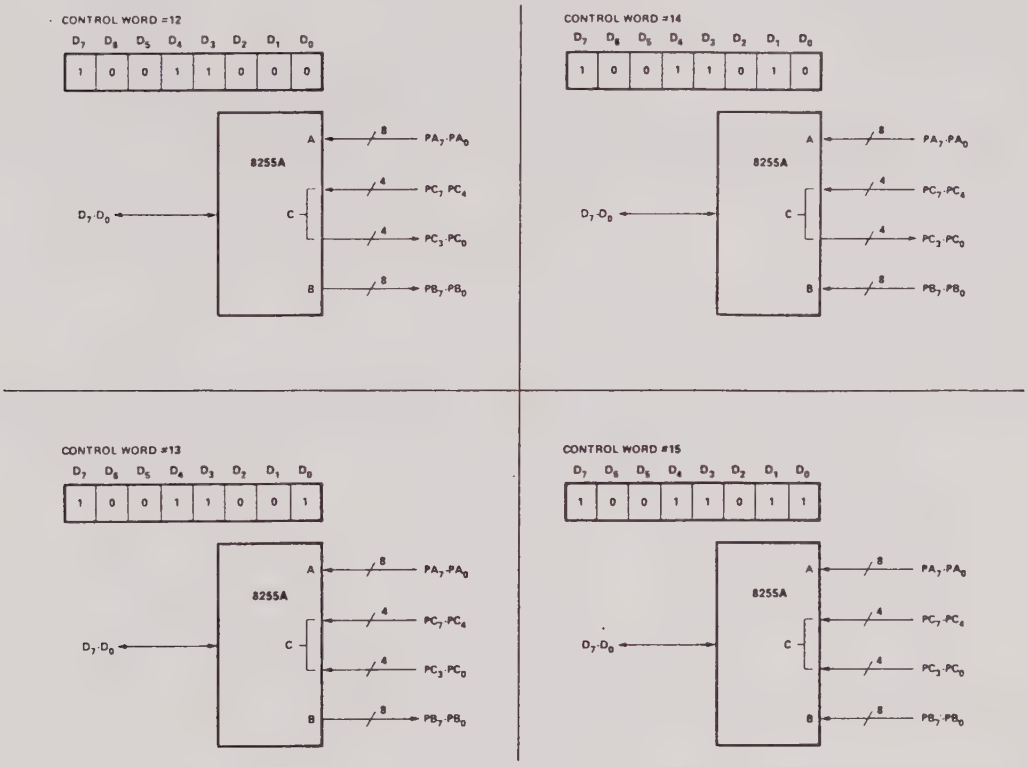


CONTROL WORD #11

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	1	1



00744A



Operating Modes

MODE 1 (Strobed Input/Output). This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "handshaking" signals. In mode 1, port A and Port B use the lines on port C to generate or accept these "handshaking" signals.

Mode 1 Basic Functional Definitions:

- Two Groups (Group A and Group B)
- Each group contains one 8-bit data port and one 4-bit control/data port.
- The 8-bit data port can be either input or output. Both inputs and outputs are latched.
- The 4-bit port is used for control and status of the 8-bit data port.

00744A

Input Control Signal Definition

STB (Strobe Input). A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F)

A "high" on this output indicates that the data has been loaded into the input latch; in essence, an acknowledgement. IBF is set by STB input being low and is reset by the rising edge of the RD input.

INTR (Interrupt Request)

A "high" on this output can be used to interrupt the CPU when an input device is requesting service. INTR is set by the $\overline{STB}$ is a "one", IBF is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{RD}$. This procedure allows an input device to request service from the CPU by simply strobing its data into the port.

INTE A

Controlled by bit set/reset of PC_4.

INTE B

Controlled by bit set/reset of PC_2.

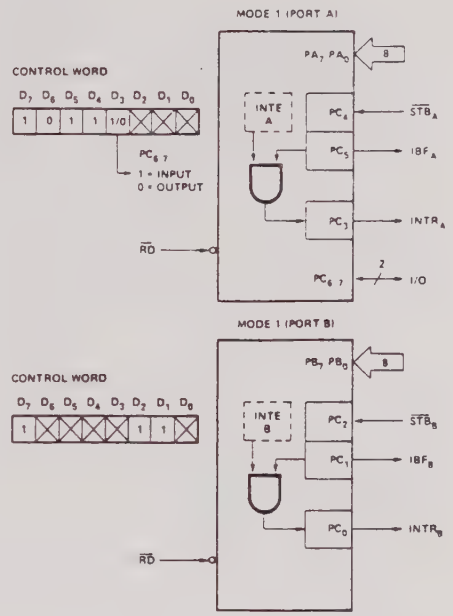


Figure 6. MODE 1 Input

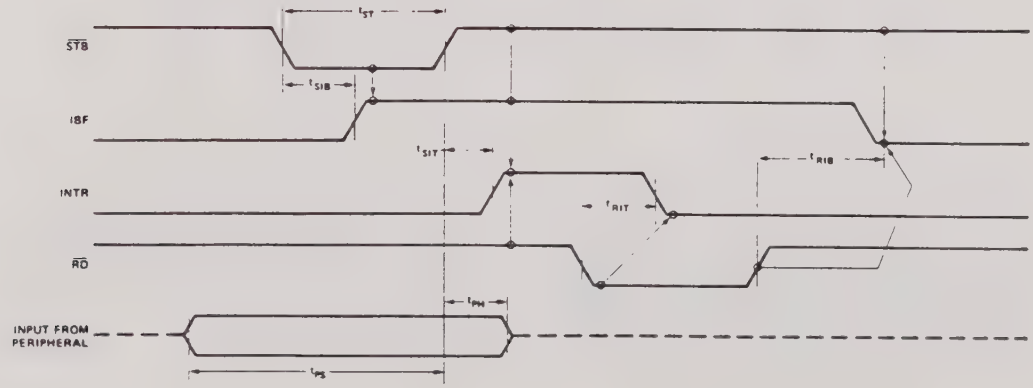


Figure 7. MODE 1 (Strobed Input)

Output Control Signal Definition

$\overline{OBF}$ (Output Buffer Full F/F). The OBF output will go "low" to indicate that the CPU has written data out to the specified port. The OBF F/F will be set by the rising edge of the WR input and reset by ACK Input being low.

$\overline{ACK}$ (Acknowledge Input). A "low" on this input informs the 8255A that the data from port A or port B has been accepted. In essence, a response from the peripheral device indicating that it has received the data output by the CPU.

INTR (Interrupt Request). A "high" on this output can be used to interrupt the CPU when an output device has accepted data transmitted by the CPU. INTR is set when ACK is a "one", OBF is a "one" and INTE is a "one". It is reset by the falling edge of WR.

INTE A

Controlled by bit set/reset of PC_6.

INTE B

Controlled by bit set/reset of PC_2.

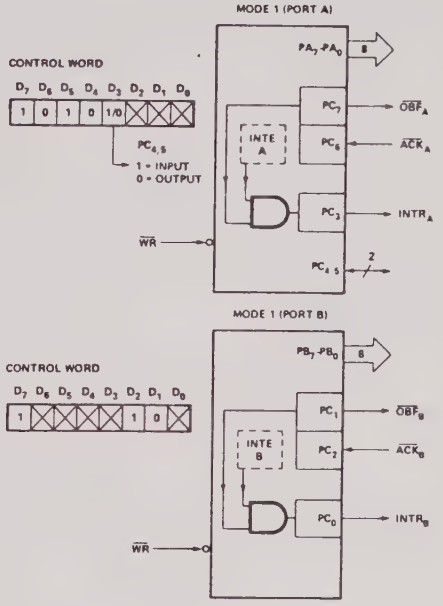


Figure 8. MODE 1 Output

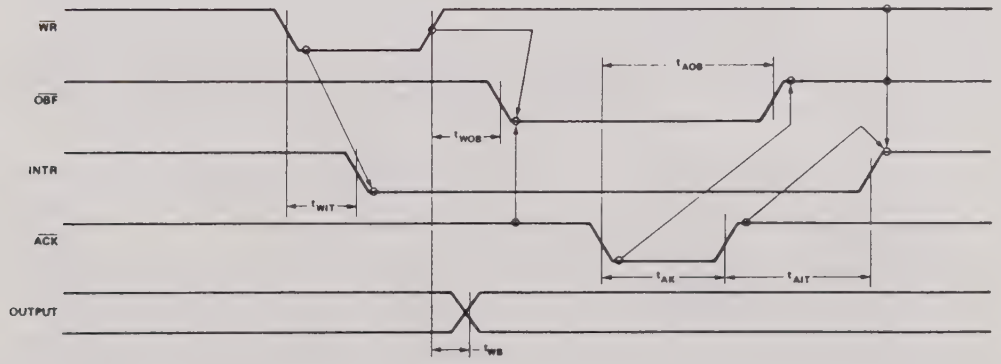


Figure 9. Mode 1 (Strobed Output)

00744A

Combinations of MODE 1

Port A and Port B can be individually defined as input or output in Mode 1 to support a wide variety of strobed I/O applications.

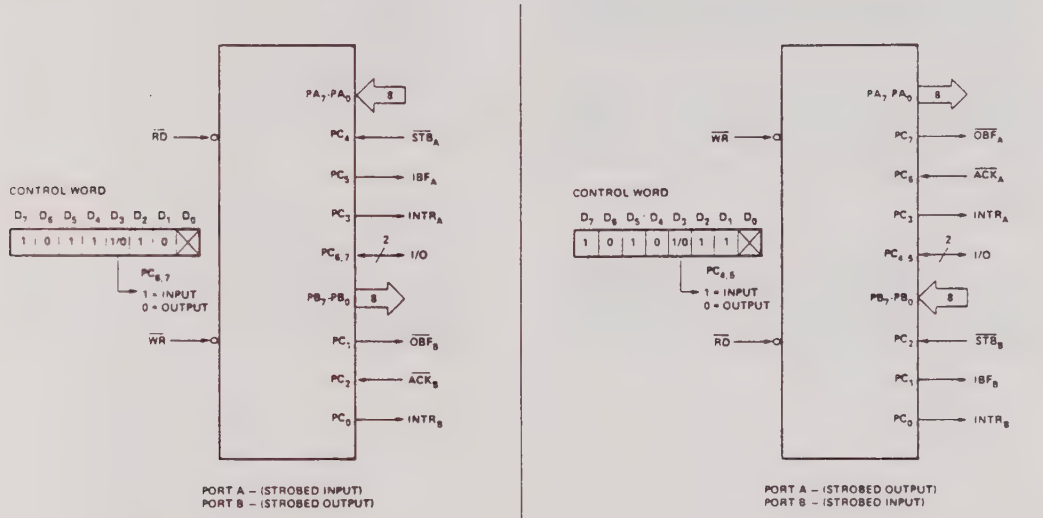


Figure 10. Combinations of MODE 1

Operating Modes

MODE 2 (Strobed Bidirectional Bus I/O). This functional configuration provides a means for communicating with a peripheral device or structure on a single 8-bit bus for both transmitting and receiving data (bidirectional bus I/O). "Handshaking" signals are provided to maintain proper bus flow discipline in a similar manner to MODE 1. Interrupt generation and enable/disable functions are also available.

MODE 2 Basic Functional Definitions:

- Used in Group A only.
- One 8-bit, bi-directional bus Port (Port A) and a 5-bit control Port (Port C).
- Both inputs and outputs are latched.
- The 5-bit control port (Port C) is used for control and status for the 8-bit, bi-directional bus port (Port A).

Bidirectional Bus I/O Control Signal Definition

INTR (Interrupt Request). A high on this output can be used to interrupt the CPU for both input or output operations.

Output Operations

$\overline{OBF}$ (Output Buffer Ful). The OBF output will go "low" to indicate that the CPU has written data out to port A.

$\overline{ACK}$ (Acknowledge). A "low" on this input enables the tri-state output buffer of port A to send out the data. Otherwise, the output buffer will be in the high impedance state.

INTE 1 (The INTE Flip-Flop Associated with OBF). Controlled by bit set/reset of PC_6.

Input Operations

$\overline{STB}$ (Strobe Input)

STB (Strobe Input). A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F). A "high" on this output indicates that data has been loaded into the input latch.

INTE 2 (The INTE Flip-Flop Associated with IBF). Controlled by bit set/reset of PC_4.

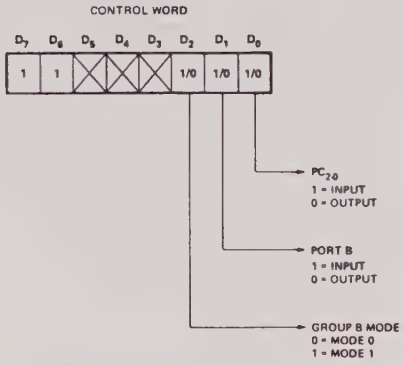


Figure 11. MODE Control Word

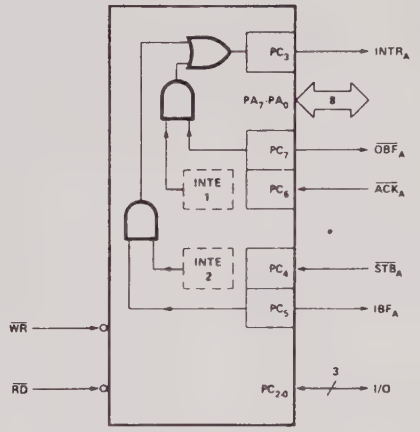


Figure 12. MODE 2

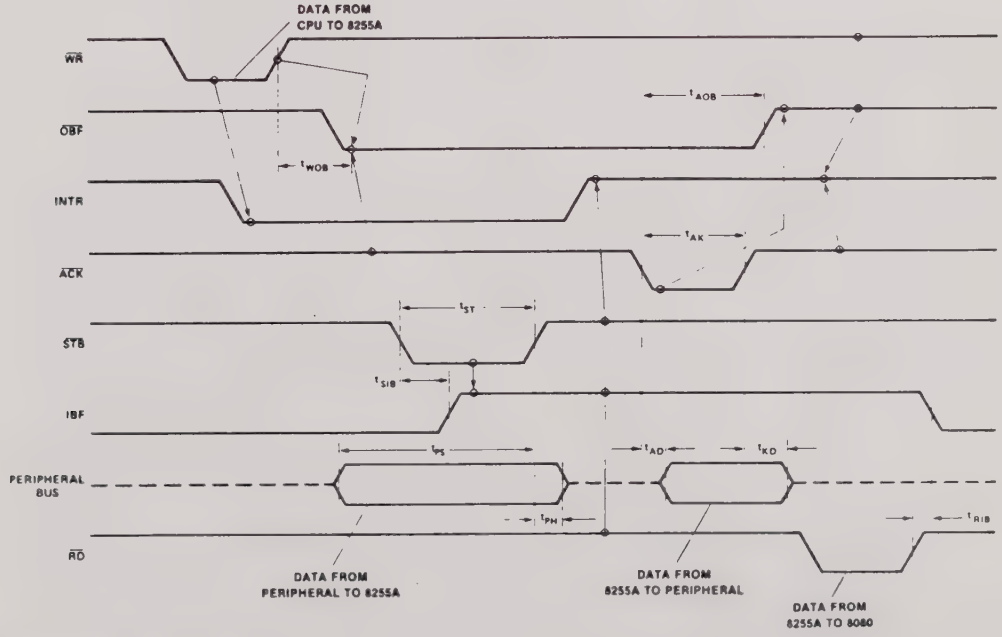


Figure 13. MODE 2 (Bidirectional)

NOTE: Any sequence where $\overline{WR}$ occurs before $\overline{ACK}$ and $\overline{STB}$ occurs before $\overline{RD}$ is permissible.
(INTR = IBF • MASK • $\overline{STB}$ • $\overline{RD}$ + $\overline{OBF}$ • MASK • $\overline{ACK}$ • $\overline{WR}$)

00744A

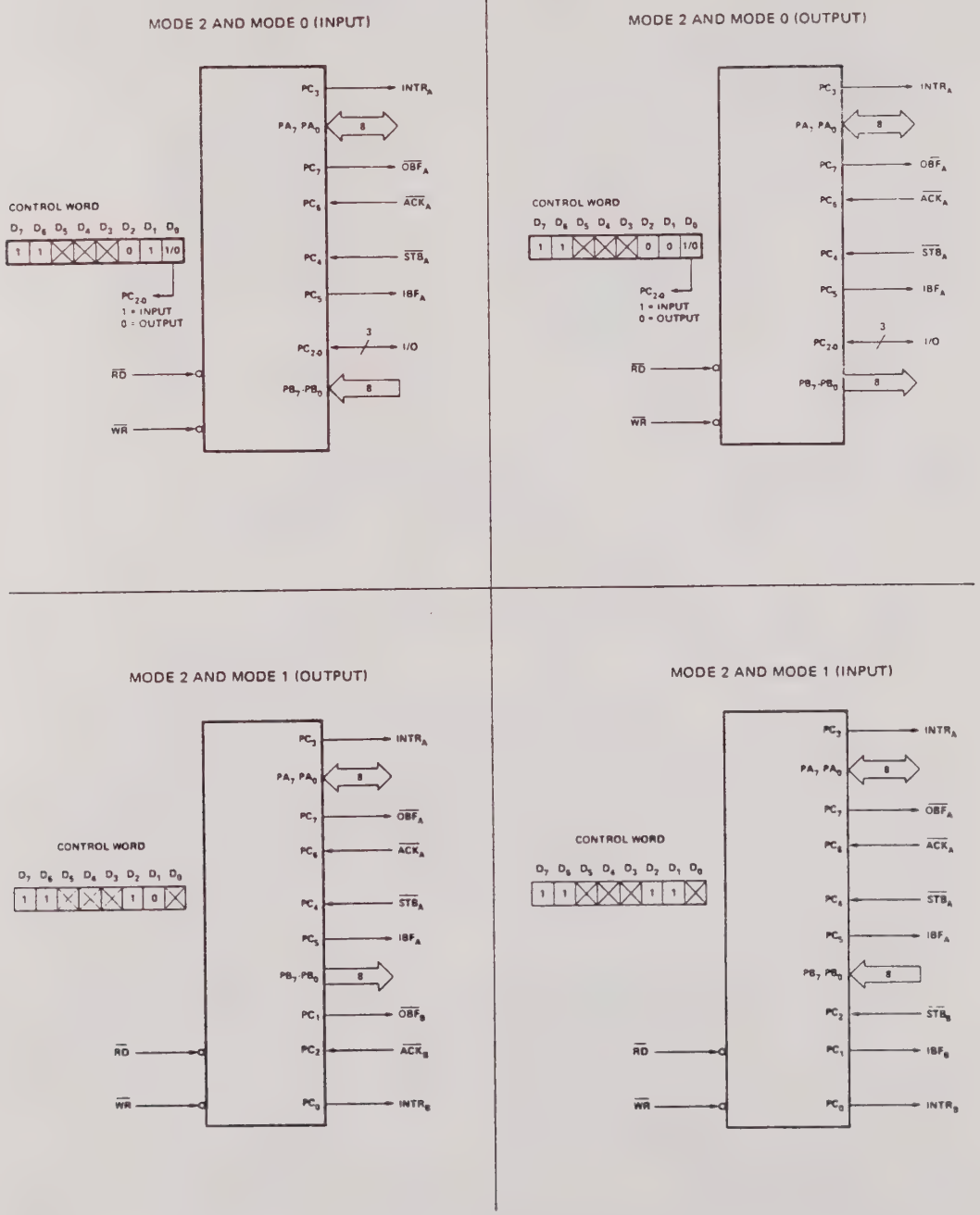


Figure 14. MODE 2 Combinations

00744A

Mode Definition Summary

	MODE 0		MODE 1		MODE 2	
	IN	OUT	IN	OUT	GROUP A ONLY	
PA_0	IN	OUT	IN	OUT	←→	
PA_1	IN	OUT	IN	OUT	←→	
PA_2	IN	OUT	IN	OUT	←→	
PA_3	IN	OUT	IN	OUT	←→	
PA_4	IN	OUT	IN	OUT	←→	
PA_5	IN	OUT	IN	OUT	←→	
PA_6	IN	OUT	IN	OUT	←→	
PA_7	IN	OUT	IN	OUT	←→	
PB_0	IN	OUT	IN	OUT	—	MODE 0 OR MODE 1 ONLY
PB_1	IN	OUT	IN	OUT	—	
PB_2	IN	OUT	IN	OUT	—	
PB_3	IN	OUT	IN	OUT	—	
PB_4	IN	OUT	IN	OUT	—	
PB_5	IN	OUT	IN	OUT	—	
PB_6	IN	OUT	IN	OUT	—	
PB_7	IN	OUT	IN	OUT	—	
PC_0	IN	OUT	$INTR_B$	$INTR_B$	I/O	
PC_1	IN	OUT	IBF_B	$\overline{OBF}_B$	I/O	
PC_2	IN	OUT	$\overline{STB}_B$	$\overline{ACK}_B$	I/O	
PC_3	IN	OUT	$INTR_A$	$INTR_A$	$INTR_A$	
PC_4	IN	OUT	$\overline{STB}_A$	I/O	$\overline{STB}_A$	
PC_5	IN	OUT	IBF_A	I/O	IBF_A	
PC_6	IN	OUT	I/O	$\overline{ACK}_A$	$\overline{ACK}_A$	
PC_7	IN	OUT	I/O	$\overline{OBF}_A$	$\overline{OBF}_A$	

Special Mode Combination Considerations

There are several combinations of modes when not all of the bits in Port C are used for control or status. The remaining bits can be used as follows:

If Programmed as Inputs –
All input lines can be accessed during a normal Port C read.

If Programmed as Outputs –
Bits in C upper (PC_7-PC_4) must be individually accessed using the bit set/reset function.

Bits in C lower (PC_3-PC_0) can be accessed using the bit set/reset function or accessed as a threesome by writing into Port C.

Source Current Capability on Port B and Port C

Any set of eight output buffers, selected randomly from Ports B and C can source 1mA at 1.5 volts. This feature allows the 8255 to directly drive Darlington type drivers and high-voltage displays that require such source current.

Reading Port C Status

In Mode 0, Port C transfers data to or from the peripheral device. When the 8255 is programmed to function in Modes 1 or 2, Port C generates or accepts "hand-shaking" signals with the peripheral device. Reading the contents of Port C allows the programmer to test or verify the "status" of each peripheral device and change the program flow accordingly.

There is no special instruction to read the status information from Port C. A normal read operation of Port C is executed to perform this function.

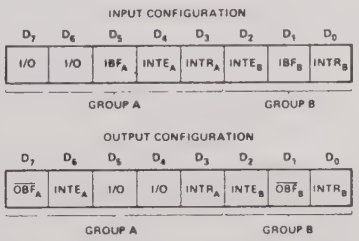

INPUT CONFIGURATION
D7 D6 D5 D4 D3 D2 D1 D0
I/O I/O IBFA INTEA INTRA INTEB IBFB INTRB
GROUP A
GROUP B
OUTPUT CONFIGURATION
D7 D6 D5 D4 D3 D2 D1 D0
OBFA INTEA I/O I/O INTRA INTEB OBFB INTRB
GROUP A
GROUP B

Figure 15. MODE 1 Status Word Format

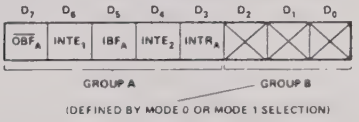


Figure 16. MODE 2 Status Word Format

00744A

APPLICATIONS OF THE 8255A

The 8255A is a very powerful tool for interfacing peripheral equipment to the microcomputer system. It represents the optimum use of available pins and is flexible enough to interface almost any I/O device without the need for additional external logic.

Each peripheral device in a microcomputer system usually has a "service routine" associated with it. The routine manages the software interface between the device and the CPU. The functional definition of the 8255A is programmed by the I/O service routine and becomes an extension of the system software. By examining the I/O devices interface characteristics for both data transfer and timing, and matching this information to the examples and tables in the detailed operational description, a control word can easily be developed to initialize the 8255A to exactly "fit" the application. Figures 17 through 23 present a few examples of typical applications of the 8255A.

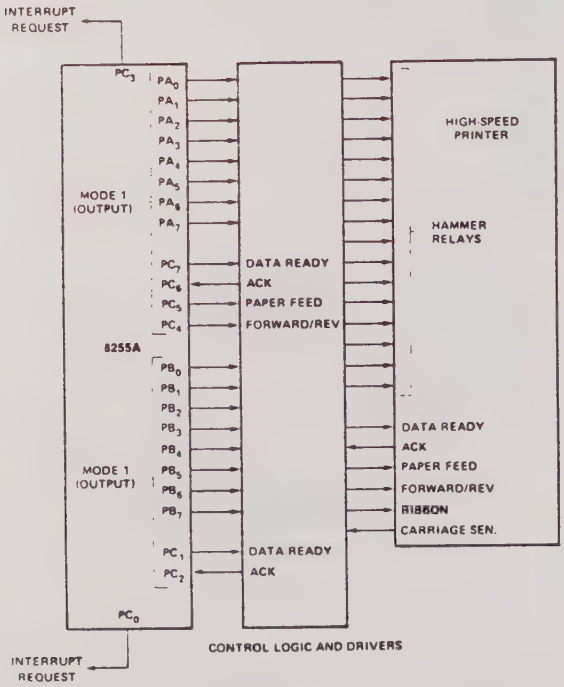


Figure 17. Printer Interface

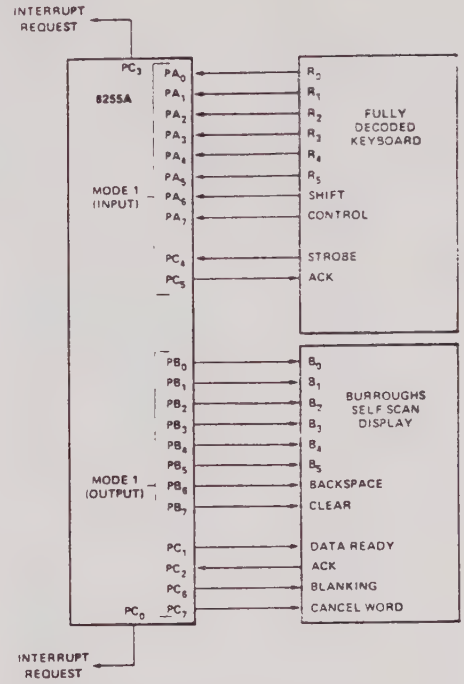


Figure 18. Keyboard and Display Interfacs

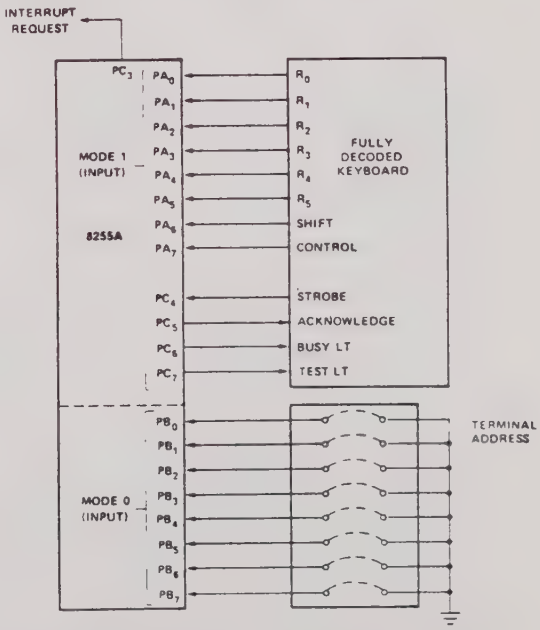


Figure 19. Keyboard and Terminal Address Interface

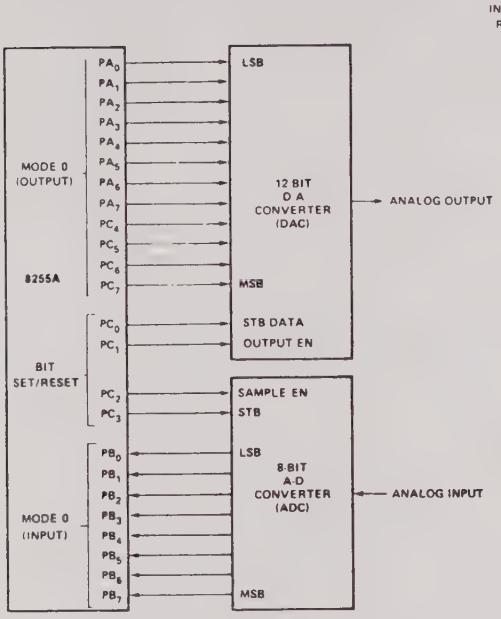


Figure 20. Digital to Analog, Analog to Digital

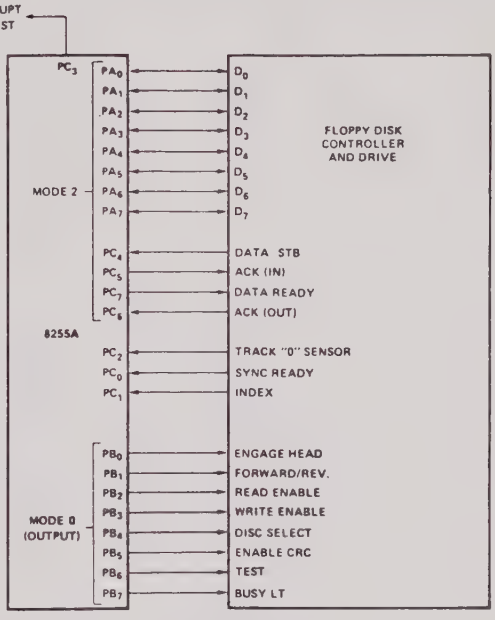


Figure 22. Basic Floppy Disc Interface

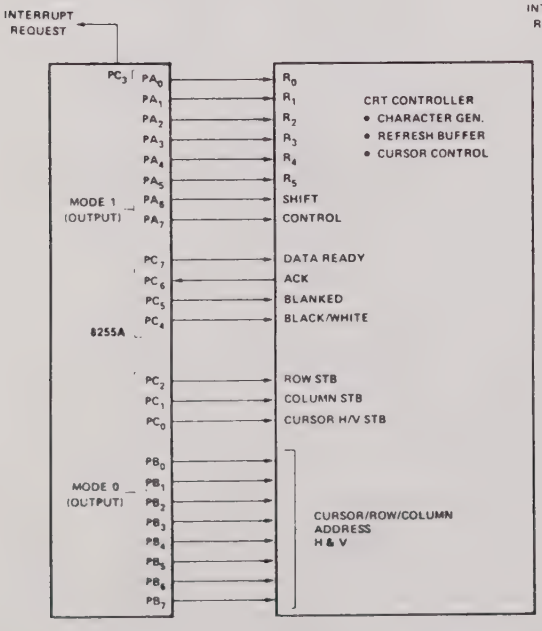


Figure 21. Basic CRT Controller Interface

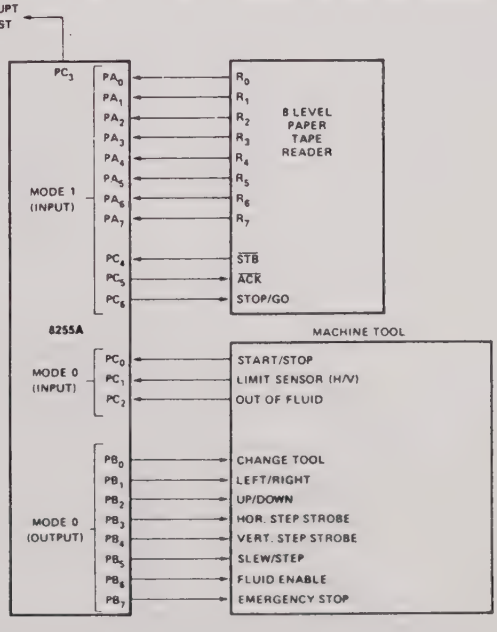


Figure 23. Machine Tool Controller Interface

00744A

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias. . . . . 0°C to 70°C
Storage Temperature . . . . . –65°C to +150°C
Voltage on Any Pin
With Respect to Ground . . . . . –0.5V to +7V
Power Dissipation . . . . . 1 Watt

COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

D.C. CHARACTERISTICS

T_A = 0°C to 70°C, V_{CC} = +5V ±5%; GND = 0V

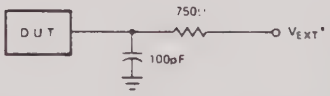
SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
V_{IL}	Input Low Voltage	-0.5	0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC}	V	
V_{OL} (DB)	Output Low Voltage (Data Bus)		0.45	V	I_{OL} = 2.5mA
V_{OL}(PER)	Output Low Voltage (Peripheral Port)		0.45	V	I_{OL} = 1.7mA
V_{OH}(DB)	Output High Voltage (Data Bus)	2.4		V	I_{OH} = -400μA
V_{OH}(PER)	Output High Voltage (Peripheral Port)	2.4		V	I_{OH} = -200μA
I_{DAR}[1]	Darlington Drive Current	-1.0	-4.0	mA	R_{EXT} = 750Ω; V_{EXT} = 1.5v
I_{CC}	Power Supply Current		120	mA	
I_{IL}	Input Load Current		±10	μA	V_{IN} = V_{CC} to 0V
I_{OFL}	Output Float Leakage		±10	μA	V_{OUT} = V_{CC} to 0V

Note 1: Available on any 8 pins from Port B and C.

CAPACITANCE

T_A = 25°C; V_{CC} = GND = 0V

SYMBOL	PARAMETER	MIN.	TYP.	MAX.	UNIT	TEST CONDITIONS
C_{IN}	Input Capacitance			10	pF	fc = 1MHz
$C_{I/O}$	I/O Capacitance			20	pF	Unmeasured pins returned to GND



*V_{EXT} is set at various voltages during testing to guarantee the specification.

Figure 24. Test Load Circuit (for dB)

00744A

A.C. CHARACTERISTICS

T_A = 0°C to 70°C; V_{CC} = +5V ±5%; GND = 0V

NOTE:
The 8255A-5 specifications are not final. Some parametric limits are subject to change.

Bus Parameters

Read:

SYMBOL	PARAMETER	8255A		8255A-5		UNIT
		MIN.	MAX.	MIN.	MAX.	
t_{AR}	Address Stable Before READ	0		0		ns
t_{RA}	Address Stable After READ	0		0		ns
t_{RR}	READ Pulse Width	300		300		ns
t_{RD}	Data Valid From READ[1]		250		200	ns
t_{DF}	Data Float After READ	10	150	10	100	ns
t_{RV}	Time Between READs and/or WRITEs	850		850		ns

Write:

SYMBOL	PARAMETER	8255A		8255A-5		UNIT
		MIN.	MAX.	MIN.	MAX.	
t_{AW}	Address Stable Before WRITE	0		0		ns
t_{WA}	Address Stable After WRITE	20		20		ns
t_{WW}	WRITE Pulse Width	400		300		ns
t_{DW}	Data Valid to WRITE (T.E.)	100		100		ns
t_{WD}	Data Valid After WRITE	30		30		ns

Other Timings:

SYMBOL	PARAMETER	8255A		8255A-5		UNIT
		MIN.	MAX.	MIN.	MAX.	
t_{WB}	WR = 1 to Output[1]		350		350	ns
t_{IR}	Peripheral Data Before RD	0		0		ns
t_{HR}	Peripheral Data After RD	0		0		ns
t_{AK}	ACK Pulse Width	300		300		ns
t_{ST}	STB Pulse Width	500		500		ns
t_{PS}	Per. Data Before T.E. of STB	0		0		ns
t_{PH}	Per. Data After T.E. of STB	180		180		ns
t_{AD}	ACK = 0 to Output[1]		300		300	ns
t_{KD}	ACK = 1 to Output Float	20	250	20	250	ns
t_{WOB}	WR = 1 to OBF = 0[1]		650		650	ns
t_{AOB}	ACK = 0 to OBF = 1[1]		350		350	ns
t_{SIB}	STB = 0 to IBF = 1[1]		300		300	ns
t_{RIB}	RD = 1 to IBF = 0[1]		300		300	ns
t_{RIT}	RD = 0 to INTR = 0[1]		400		400	ns
t_{SIT}	STB = 1 to INTR = 1[1]		300		300	ns
t_{AIT}	ACK = 1 to INTR = 1[1]		350		350	ns
t_{WIT}	WR = 0 to INTR = 0[1]		850		850	ns

Notes: 1. Test Conditions: 8255A: C_L = 100pF; 8255A-5: C_L = 150pF.
2. Period of Reset pulse must be at least 50μs during or after power on. Subsequent Reset pulse can be 500 ns min.

00744A

Figure 25. Input Waveforms for A.C. Tests

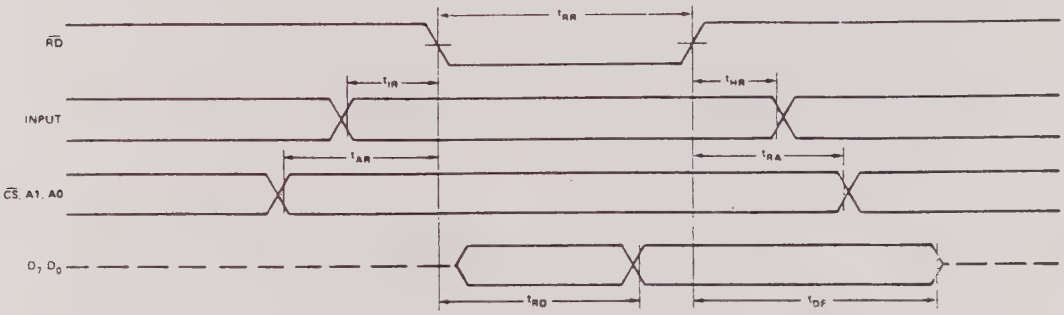


Figure 26. MODE 0 (Basic Input)

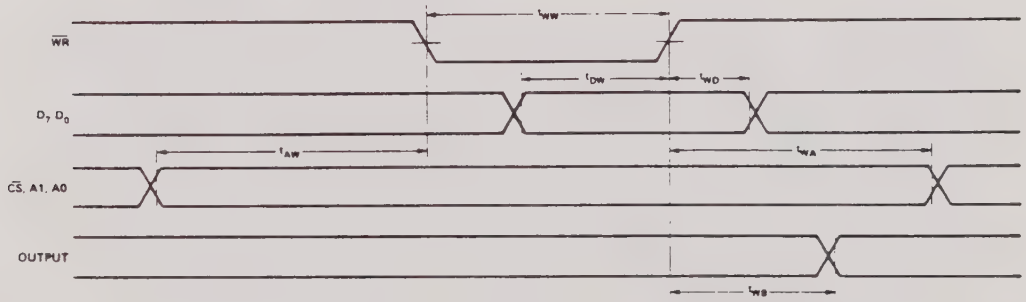


Figure 27. MODE 0 (Basic Output)

GenRad
Development Systems Division

Technical Publications Remarks Form

Please use this form to submit your suggestions for revisions, corrections, or additions to this publication. Your comments will be promptly investigated by appropriate technical personnel, and action will be taken as required. If your answer to any of the questions is 'NO' or requires qualification, please include any additional comments on a separate sheet and enclose it inside this pre-addressed form.

TITLE: _____

ORDER NO. _____

DATED _____

	YES	NO
DOES THIS DOCUMENT MEET YOUR NEEDS?	_____	_____
IS THE MATERIAL CONTAINED IN THIS DOCUMENT:		
ACCURATE?	_____	_____
EASY TO READ AND UNDERSTAND?	_____	_____
ORGANIZED FOR CONVENIENT REFERENCING?	_____	_____
WELL ILLUSTRATED WITH USEFUL EXAMPLES?	_____	_____
COMPLETE?	_____	_____

ERRORS IN THE PUBLICATION

SUGGESTIONS FOR IMPROVEMENT

(Please Print)

FROM: NAME _____ DATE _____

ORGANIZATION _____

ADDRESS _____

TITLE _____

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 608 CULVER CITY, CA

POSTAGE WILL BE PAID BY ADDRESSEE

GenRad
Development Systems Division

5730 Buckingham Parkway
Culver City, CA 90230

ATTN: MARKETING TECHNICAL PUBLICATIONS